(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 404 337 B1

(12) EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention of the grant of the patent:
24.03.1999 Bulletin 1999/12

(51) Int Cl.[6]: **H04L 12/28**

(21) Application number: **90305289.2**

(22) Date of filing: **16.05.1990**

(54) High-speed mesh connected local area network

Lokales Maschennetzwerk hoher Geschwindigkeit

Réseau maillé local à grande vitesse

(84) Designated Contracting States:
AT BE CH DE DK ES FR GB GR IT LI LU NL SE

(30) Priority: **22.06.1989 US 370285**

(43) Date of publication of application:
27.12.1990 Bulletin 1990/52

(60) Divisional application: **97201068.0 / 0 817 424**

(73) Proprietor: **DIGITAL EQUIPMENT CORPORATION**
Maynard, Massachusetts 01754 (US)

(72) Inventors:
- **Schroeder, Michael D.**
  Cupertino, California 95014 (US)
- **Needham, Roger M.**
  Cambridge CB3 7PY (GB)
- **Birrell, Andrew D.**
  Los Altos, California 94022 (US)
- **Rodeheffer, Thomas L.**
  Mountain View, California 94040 (US)
- **Murray, Hallam G.,Jr.**
  Menlo Park California 94025 (US)
- **Thacker, Charles P.**
  Palo Alto, California 94301 (US)
- **Satterthwaite, Edwin H.,Jr.**
  Palo Alto, California 94306 (US)

(74) Representative: **Goodman, Christopher et al**
Eric Potter Clarkson,
Park View House,
58 The Ropewalk
Nottingham NG1 5DD (GB)

(56) References cited:
EP-A- 0 274 709 **US-A- 4 466 060**
US-A- 4 701 756

- **IEEE COMMUNICATIONS MAGAZINE. vol. 22, no. 8, August 1984, US pages 36 - 40 C. PETITPIERRE 'MESHED LOCAL COMPUTER NETWORKS'**
- **COMPUTER NETWORKS. vol. 3, 1979, AMSTERDAM NL pages 267 - 286 P.KERMANI ET AL 'VIRTUAL CUT-THROUGH: A NEW COMPUTER COMMUNICATION SWITCHING TECHNIQUE'**
- **IEEE TRANSACTIONS ON COMPUTERS. vol. C-36, no. 5, May 1987, NEW YORK US pages 547 - 553 W.J.DALLY ET AL 'DEADLOCK-FREE MESSAGE ROUTING IN MULTIPROCESSOR INTERCONNECTION NETWORKS'**
- **IEEE GLOBAL TELECOMMUNICATIONS CONFERENCE vol. 2, 15 November 1987, TOKYO, JP pages 1410 - 1414 T.SUDA ET AL 'TREE LANS WITH COLLISION AVOIDANCE: PROTOCOL AND SWITCH ARCHITECTURE'**
- **COMPUTER COMMUNICATION REVIEW. vol. 18, no. 4, August 1988, NEW YORK US pages 330 - 338 C.CHENG ET AL 'A PROTOCOL TO MAINTAIN A MINIMUM SPANNING TREE IN A DYNAMIC TOPOLOGY'**
- **IEEE TRANSACTIONS ON SOFTWARE ENGINEERING. vol. 13, no. 3, March 1987, NEW YORK US pages 398 - 405 Y.K.DALAL 'A DISTRIBUTED ALGORITHM FOR CONSTRUCTING MINIMAL SPANNING TREES'**

Remarks:
Divisional application 97201068.0 filed on 10/04/97.

EP 0 404 337 B1

Printed by Jouve, 75001 PARIS (FR)

Description

[0001] The present invention relates generally to computer communications networks for interconnecting computers and particularly to a mesh connected local area network for routing information packets between computers.

BACKGROUND OF THE INVENTION

[0002] Local area networks (LANs) are commonly used to transmit messages between relatively closely located computers. Referring to Figures 1A, 1B and 2, there are at least three basic types of organizational architectures for LANs: linear (shown in Figure 1A), ring (shown in Figure 1B), and mesh (shown in Figure 2). Ethernet, for example, is a widely used linear LAN for interconnecting computer workstations, mainframes, and minicomputers.

[0003] For the purposes of this discussion linear LANs are defined to be single channel LANs in which message packets are broadcast so as be heard by all hosts (H) on the network, although usually only the host that is addressed by a packet will choose to listen to it.

[0004] The present invention solves the primary problems which have heretofore prevented mesh connected LANs from providing reliable high speed communications among a large number of interconnected host computers. For the purposes of this discussion, "a mesh connected network" means a network of switches connected in an arbitrary topology.

[0005] Before explaining the significance of the problems solved by the present invention, we will briefly consider the differences between mesh connected local area networks and linear and ring networks, and the motivations for building mesh connected networks even though such networks are generally more expensive and complicated than linear and ring LANs.

[0006] Linear and ring LANs have the advantage of architectural simplicity and well known solutions to most of the problems required for successful commercial application - and have well established records of reliability. However, linear and ring LANs have at least two major technological limitations - both the number of hosts (i.e., workstations and other computers) and the quantity of data that can be transmitted through such LANs are limited by the availability of only a single data transmission path. As more and more hosts are added to a linear or ring LAN, the amount of traffic on the single data path will increase and the average amount of time that each host must wait to send a message will also increase. Eventually, if enough hosts share a single LAN the delays will become unacceptable.

[0007] It can be shown that simply increasing the rate of data transmission on linear and ring LANs does not completely solve the problem of network congestion because some of the delays in such networks are related to the length of time that it takes for a message to traverse the length of the network - i.e., some delays are proportional to the physical length of the network, regardless of the rate of data transmission.

[0008] For instance, it has been shown that the maximum usable data transmission rate in linear LANs is inversely proportional to the physical length of the network's channel. As a result, it would appear that useful linear LANs cannot use data transmission rates much higher the 10 Megabaud rate currently used by Ethernet - because the use of substantially higher data rates will restrict the length of the network. In addition, linear LANs have the problem that, since only one data packet can be sent at a time, there must be a mechanism for deciding who (i.e., which host on the LAN) will have control of the LAN at any one time. A simple consideration of signal speed limitations imposed by the speed of light indicates that the length of linear LANs must be fairly limited (e.g., to several kilometers), and that network performance will degrade as more hosts are added to the LAN because of contention for control of the LAN.

[0009] While ring LANs can run at arbitrarily high data rates, rings LANs suffer from high latency - the delay between transmission and receipt of a message, which is proportional to the length of the network and the number of nodes which must be traversed. Ring LANs are also not very fault tolerant, and are very limited in terms of their configuration.

[0010] While the above noted problems with linear and ring LANs have not overly hampered their usefulness so far, the growing need for LANs with hundreds of hosts and for data transmission rates in the range of 100 Megabits per second exceeds the capability of the presently existing linear and ring LANs.

[0011] The primary advantage of using a mesh connected LAN is the availability of many parallel communications paths. This allows the simultaneous transmission of messages between different pairs of network hosts. Thus a mesh connected network can achieve much higher bandwidth than a comparable linear or ring network - because the throughput of the network is not limited by the throughput limitations of the network's links.

[0012] Another advantage of mesh connected networks over ring LANs is that mesh networks can have relatively low latency. Latency is generally proportional to the number of nodes that must receive and retransmit a message packet. A well designed mesh LAN can have a relatively small number of nodes between any selected pair of hosts in comparison to a ring LAN with a similar number of hosts.

[0013] Another advantage of mesh connected networks is that a well designed mesh connected network will provide several potential communication paths between any selected pair of hosts, thereby reducing the amount of time that hosts must wait, on average, before transmitting a message. In other words, contention for use of the network can be

greatly reduced because many hosts can use the network simultaneously.

[0014] Traditionally, while mesh networks have been discussed in computer science literature and a few patents, mesh networks have never achieved commercial success due to several well known and relatively intractable problems. In particular, the most difficult problems have been (1) deadlock, (2) handling broadcast messages, (3) how to reconfigure the network when a network component fails, and (4) how to organize the routing of messages through the network so that the network throughput exceeds the throughput of a single link. These problems, and their solutions by the present invention are described below.

[0015] EP-0274709 and Computer Networks, vol. 3, pp.267-286; Kermani *et al:* "Virtual Cut-Through: A New Computer Communication Switching Technique" describe cut-through routing in a mesh connected network.

[0016] IEEE Communications Magazine, vol. 22, no. 8, pp. 36-40; Petitpierre: "Meshed Local Computer Networks" describes meshed network topologies with point-to-point full-duplex links.

[0017] IEEE Transactions on Computers, vol C-36, no. 5, pp. 547-553; Dally *et al:* "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks" teaches that deadlocks in a cut-through network can be avoided by ensuring that there are no cycles in a channel dependency graph.

SUMMARY OF THE INVENTION

[0018] The present invention provides a mesh connected local area network according to claim 1 and a method of operating a mesh connected local area network according to claim 9. The mesh connected network has high host-to-host bandwidth, low host-to-host latency, and high aggregate bandwidth. The mesh connected network consists of a number of interconnected switches which are coupled, in turn, to the hosts that are members of the local network. The switches are cut-through, nonblocking switches that are coupled to each other and to the hosts by a multiplicity of point to point links.

[0019] The switches are organized as a spanning tree with one switch being denoted the root node of the tree. Using a node ranking rule which will be described below, every switch is ranked in terms of how "close" it is to the root node.

[0020] Every link in the network is denoted as an "up" link in one direction and as a "down" link in the other direction. The up direction is the one for which the switch at one end of the link is closer to the root than the switch at the other end of the link.

[0021] In addition, each switch has a routing mechanism for automatically routing a received message packet toward its target host. In particular, the routing mechanism of the present invention allows numerous packets to be routed simultaneously through the network, and prevents deadlock by ensuring that all message packets follow a sequence of zero or more up links, followed by zero or more down links. No up links are traversed after the message packet has been routed down even a single down link.

[0022] High aggregate bandwidth is achieved by simultaneously routing many data packets through the network. Low latency is achieved, in part, by providing switches which start retransmitting (i.e., forwarding) packets well before receiving the ends of those packets. This is knows as cut-through switching.

[0023] A packet buffering scheme prevents node starvation and enables the routing of broadcast messages. In addition, the flow control and data buffering of the present invention compensates for any mismatches between the clock rates of neighboring switches.

BRIEF DESCRIPTION OF THE DRAWINGS

[0024] Additional objects and features of the invention will be more readily apparent from the following detailed description and appended claims when taken in conjunction with the drawings, in which:

[0025] Figure 1A is a block diagram of a linear local area network, and Figure 1B is a block diagram of a ring local area network.

[0026] Figure 2 is a block diagram of a small mesh connected local area network in accordance with the present invention.

[0027] Figure 3 is a more detailed diagram of a section of a local area network in accordance with the present invention.

[0028] Figure 4 depicts an example of deadlock in a mesh connected LAN.

[0029] Figure 5 is a conceptual diagram of the concept of up and down links in a mesh connected LAN.

[0030] Figure 6 is a timing diagram depicting the transmission of a data packet and the corresponding flow control signals.

[0031] Figure 7 is a block diagram of a network controller for one host computer.

[0032] Figure 8 is a block diagram of the switch used in the preferred embodiment.

[0033] Figure 9 is a block diagram of the crossbar switch used in the preferred embodiment.

[0034] Figure 10 is a block diagram of the data flow control circuitry for a chain of connected network members.

[0035] Figure 11 is a block diagram of two connected link units in a switch.

[0036] Figure 12 is a detailed block diagram of a link unit.

[0037] Figure 13 is a block diagram of the router used in the switch of Figure 8.

[0038] Figure 14 schematically depicts the process of selecting a link vector from a routing table using the network address as part of the lookup address.

[0039] Figure 15 is a block diagram of the route selection mechanism of the router in Figure 13.

[0040] Figure 16 is a timing diagram for the router of Figure 13.

[0041] Figure 17 depicts a mesh network as a spanning tree.

DESCRIPTION OF THE PREFERRED EMBODIMENT

[0042] Figure 2 shows a conceptual representation of a mesh connected local area network 100 in accordance with the present invention, although many important features of the present invention are not shown in this Figure. Unlike the prior art mesh networks, there is no particular hierarchy of nodes and no requirements as to how the nodes of the network are interconnected. The nodes of the network could be randomly interconnected and the network would still function properly, although a well thought out set of interconnections will provide somewhat better performance.

[0043] In Figure 2 the host computers which use the network are labelled H, and the nodes which comprise the local area network (LAN) are called switches and are labelled S. In this conceptual diagram sixteen switches are used to interconnect about eighty hosts. It should be noted that the switches S are multiported, cut-through nonblocking switches which can simultaneously couple a multiplicity of incoming links to various selected outgoing links. These switches enable numerous data packets to be simultaneously routed through the network.

GLOSSARY

[0044] To clarify the following discussion, the following definitions are provided.

[0045] "Channel" is the term used to refer to one half of a link, as defined below. In general, each channel is a single direction communication channel for transmitting data packets between two members of a network. In some contexts a channel is called an "up link" or "down link" to identify the direction of data flow in the channel.

[0046] A "host" is any computer or workstation that is connected to the network and which can be used to send and receive messages. Each letter "H" in Figure 2 represents one host.

[0047] A "member of the network" or "network member" is either a host or a switch.

[0048] A "mesh connected network" is a network of switches connected in an arbitrary topology.

[0049] A "message" is any set of information or data which is transmitted from one member of the network to another. As will be explained in detail below, most messages are sent from one host to another, but occasionally, network control messages are sent from one switch to another.

[0050] "Packet", "data packet" and "message packet" all mean the basic unit of information which is transmitted through the network. Basically, any set of information that is sent through the network is first packaged into one or more packets. Each packet includes a header that specifies the destination of the packet, and a tail which declares the end of the packet. Thus a short message (e.g., less than 10,000 bytes) will be typically transmitted as a single packet, whereas a long message (e.g., a long document or data file) will be broken into a stream of consecutively transmitted packets.

[0051] "Retransmitting" a packet means forvarding a packet that has been received or partially received by a switch.

[0052] A "port" is the circuit in a switch (or host) which couples the switch (or host) to a link.

[0053] A "switch" is a physical device that is used to receive and route packets through the network. In the preferred embodiment switches can be connected to at least a dozen hosts and/or other switches. Each circle in Figure 2 labelled "S" represents one switch.

[0054] A "link" is the apparatus which physically connects any two members of the network. In Figure 2 each straight line between a host H and a switch S, or between two switches, represents a link. In the context of the present invention, each link between two network members is a full duplex, two way channel, which allows simultaneous communications in both directions. Both ends of each link are terminated by a "link circuit" which is also called a "port".

[0055] A "network address" is a value, assigned to each network member, used to index into a "routing table". The entry in the routing table specified by the network address provides information corresponding to legal routes through the network to the network member.

[0056] "Reconfiguration" is the process of determining all the legal data transmission paths for data packets being transmitted by the network. Every time that a new switch of link is added to the network, and every time that a switch or link is removed from the network or fails to work properly, a network reconfiguration takes place. An important feature of the present invention is that not all of the physical multi-link paths between two hosts are legal transmission paths.

[0057] "Spanning tree," as used herein, means a representation of the interconnections between the switches in a

mesh connected network. Technically, a spanning tree is a non-cyclic connected subgraph which represents a portion of the network, excluding the host computers and certain links between switches. The excluded links make the network an acyclic graph rather than a tree because the nodes of the spanning tree can have interconnections within each level of the graph.

[0058] A "Netlist" is a representation of the switches and links between switches in a network.

[0059] The "root", "root node" or "root switch" of a network is a switch S which is designated as the root of the spanning tree representation of the network. The root node has several special responsibilities during reconfiguration of the network, and also for retransmitting broadcast messages, i.e., messages that are sent to all of the hosts in the network.

NETWORK CONNECTIONS AND ROUTING

[0060] Referring to Figure 3, there is shown one section of a mesh connected network in accordance with the present invention. In the preferred embodiment, each host 120 in the network has a network controller 122 which couples the host 120 to two distinct switches (e.g., switches 124 and 126 in the case of host 120). The two links 128 and 130 which couple the host 120 to switches 124 and 126 are identical, except that only one of the two links is active at any one time. For this reason link 130 is shown as a dashed line to indicate that it is inactive.

[0061] Whenever the active link between a host computer and a switch fails, the host's network controller 122 automatically activates the other link 130 - thereby reconnecting the host to the network. In addition, it is strongly preferred that the two links 128 and 130 for each host be coupled to two different switches so that if an entire switch fails all the hosts coupled to that switch will have alternate paths to the network. Generally, the provision of two alternate paths or channels from each host to the network provides sufficient redundancy that no single hardware failure can isolate a host from the network.

[0062] It is noted here that each "link" between network members is actually two communications channels which simultaneously carry data in opposite directions. In the preferred embodiment, each link 128 in the network can be up to 100 meters in length when coaxial cable is used, and up to 2 kilometers miles in length when fiber optic cabling is used.

[0063] When using coaxial cable, the amount of wiring needed by the network can be reduced by using a single line of cable to simultaneously transmit signals in both directions over the link. At each end of the cable there is a transmitter and a detector. The detector regenerates the signals sent by the transmitter at the other end of the cable by subtracting the output of the transmitter at the same end of the cable from the signal received by the detector at its end of the cable. Such full duplex, single wire communication channels are well known, and are not essential to implementing the present invention.

[0064] Numerous data packets can be simultaneously transmitted through the network. For example consider the example of a first packet being sent from host 132 to host 134 while a second packet is sent from host 136 to host 138. Figure 3 shows a route P1, comprising three links coupled by two switches which can be used for sending the first packet from host 132 to host 134. Route P2 shown in Figure 3 can simultaneously be used to send the second packet from host 136 to host 138. In this example, both data packets are simultaneously routed through switch 140. This is possible because the switches used in the present invention are multiported nonblocking switches. Each switch contains a crossbar circuit which can simultaneously couple a multiplicity of incoming links to distinct outgoing links.

[0065] While packets are generally sent from one host H in the network to another host H, it is noted that during reconfiguration of the network data packets are sent to computers in the switches themselves. This aspect of data packet routing will be discussed below in the sections entitled Routing Tables and Reconfiguration Process.

Deadlock

[0066] One of the features of the present invention is that it provides a mesh connected network that cannot suffer from "deadlock". padlock in a network can be thought of as the electronic analog of gridlock at a traffic intersection. Figure 4 shows four host computers A, B, C and D and four switches S. Each host is trying to send a data packet 148 to another host that is separated from the transmitting host by two switches. The destination of each packet is denoted by the label in the box 148 which symbolizes the packet. For instance, the packet 148 being sent by host A has a destination of host C.

[0067] For the purposes of this example it is assumed that the data packets being sent are larger than the data buffers in the switches, and that therefore the data packets will occupy a chain of two or more links during the transmission of the packet. As shown in Figure 4, the progress of each data packet is blocked because the link needed for the next step of the transmission is blocked by another one of the packets.

[0068] As will be appreciated by those skilled in the art, deadlock can also occur with small data packets. In particular, the data buffer in a switch can become filled with two or more data packets, thereby preventing any more data from being sent through the link that is connected to the filled data buffer. Thus in Figure 4 each blocked packet 148 can be

replaced by a sequence of two or more packets, the first of which is being blocked because the link needed for the next step in its route is blocked by another one of the packets.

[0069] Clearly this deadlocked condition will not happen very often because it requires four hosts to initiate the sending of new data packets virtually simultaneously. However, it is unacceptable for deadlock to ever occur because it will cause the network to "crash" and messages to be lost.

Up/Down Routing

[0070] The present invention completely prevents deadlock by using a new type of routing procedure which automatically routes messages so that they will not deadlock one another. Referring again to Figure 4, the implication of the data paths shown is that one could, at least theoretically, have a "cycle" in which a data packet is sent into an endless loop through the four switches. While cycles are not, by themselves, usually a problem, the availability of data paths which form a cycle is a symptom of mesh networks which can suffer deadlock.

[0071] Referring to Figure 5, there is shown a somewhat complicated example of a ten node network of switches S1 to S10. All lines between the switches represent bidirectional links.

[0072] For reasons which will soon be explained, every link between the switches has been assigned a direction, as indicated by the arrows on the links. The arrows on the links are said to point "up" toward the root node of the network. More specifically, when a data packet is transmitted through a link in the same direction as the arrow on that link, the data packet is said to be going on an "up link" or, more correctly, an "up channel". When a data packet is transmitted through a link in the opposite direction as the arrow on that link the data packet is said to being going on a "down link" or "down channel".

[0073] The basic routing rule used in the present invention is that all legal routes for a data packet comprise zero or more up channels, followed by zero or more down channels. Once a data packet has been transmitted through a down channel it cannot be transmitted through an up channel.

[0074] The basic routing rule as just stated defines the legal routes for a packet from a "global perspective" - that is from the viewpoint of someone looking at the network as a whole. From the perspective a single switch, when a packet travels on an "up link" to the switch, that packet is received on a down link. Thus, from the "local switch perspective", packets received on "down links" can be forwarded on either an up or down link; packets received on "up links" can be forwarded only on down links.

[0075] In addition, it should be understood that for all links between hosts and switches, the up direction is toward the switch. The channel from a host computer to a switch is always an up link or channel, and the channel from a switch to a host is always a down link or channel. Thus when a host computer transmits a data packet, the first channel that the data packet goes over is always an up channel. Similarly, the last channel that a data packet goes over as it is received by a host computer is always a down channel.

[0076] The lower left portion of Figure 5 comprising switches S1, S3, S5 and S10 will now be used to show why deadlock is impossible using the up/down routing mechanism. If one tries to impose the data paths from Figure 4 onto these switches in Figure 5, one will see that all of the data paths in Figure 4 are legal except one: the data path from host B to host D through switches S3, S5 and then S10 is not legal. This is because the path from S3 to S5 is a down channel while the path from S5 to S10 is an up channel. This contradicts the rule that up channels cannot be used after down channels. The solution is that message from host B to host D must first go from S3 to S1 (which is an up channel) and then from S1 to S10 (which is a down channel).

[0077] The directionality of each link between switches in the network is determined as follows. Every switch (and host computer) is permanently assigned a unique 48-bit identifier, called the UID. Such UIDs are used in Ethernet networks to uniquely identify every member of the network. As will be discussed later, every switch in the network is assigned a 7-bit SHORT ID, and every host computer is assigned an 11-bit network address.

[0078] The first rule is that the switch with the lowest UID in the entire network is called the root node and is assigned a network level of zero. A corollary of the first rule is that all links to the root node are directed upwards toward the root.

[0079] In Figure 5 it is assumed that each switch is assigned a UID equal to its reference numeral: S1 is assigned a UID of 1, then S2 is assigned a UID of 2, and so on. Thus switch S1 is the root and the links to S1 from switches S2, S3, S9 and S10 are directed upwards toward switch S1.

[0080] The second rule is that switches are assigned network levels based on the minimum number of links between the switch and the root, and that links between switches at different network levels are directed upward toward the lower network level. For instance, switch S3 is at network level 1 and switch S8 is at network level 2, and thus the link from S8 to S3 is upward toward S3.

[0081] The third and final rule for assigning directionality to links is that links between switches at the same network level are upward toward the switch with the lower UID. Thus, since switches S2 and S3 are both at network level 1, the link between them is upward toward S2.

[0082] Another example of a legal route through the network is as follows: to send a packet from host C to host E,

the packet could go via path P3 or path P4. Path P3, which goes through switches S5, S3, S8 and S9, is legal because it follows up channels and then down channels. Path P4, which goes through switches S5, S7 and then S8, is legal because it follows two down channels.

[0083] While path P4 is shorter than path P3, path P3 might be preferred if the first link of P4 is blocked while the first link in path P3 is available. Thus, the preferred path through the network will depend on which links are already being used by other data packets, and the preferred path will not always be the shortest legal path.

[0084] An example of an illegal route for a packet being sent from host F to host G is switches S7 to S6 (down link), followed by S6 to S8 (up link). That route is illegal because it has an up link (S6 to S8) after a down link (S7 to S6) - which is not allowed. A legal route from F to G would be S7 to S5 (up), S5 to S3 (up) and S3 to S8 (down).

[0085] The above described method of assigning directionality to the links in the network, and to defining legal routes through the network has been found by the inventors to eliminate not only the deadlock problem, but to also provide a convenient mechanism for handling broadcast message packets, as will be described in detail below.

PACKET FLOW CONTROL

[0086] In order to understand many of the features of the preferred embodiment, it is first necessary to understand how "flow control" works. Referring to Figure 3, consider the example of a 16,000 byte packet being sent from host 132 to host 134. For the purposes of this example, we will assume that each switch port contains a 4k byte FIFO buffer for temporarily storing a portion of an incoming data packet.

[0087] Initially, the packet is transmitted by host 132 along path P1 to switch 142. If link 144 is already being used to transmit another data packet, the 4k buffer in switch 142 will soon overflow - unless host 132 can be instructed to temporarily halt transmission of the data packet.

[0088] In the preferred embodiment data is continuously transmitted in both directions over every link, such as link 146 between host 132 and switch 142. If there is no data which needs to be sent, then synchronization bytes are sent. Synchronization bytes are simply null data.

[0089] At the same time that data is being transmitted, flow command signals are also sent by a simple form of time multiplexing: every 256th byte that is transmitted is a flow control command. The transmission of flow commands is not synchronized with packet boundaries; a flow command is on every link once every 256 byte cycles regardless of what data the link may be carrying. Thus if a 700 byte message were being sent over a link, the data stream representing the message might look like this: the first 200 bytes of the message, followed by a one-byte flow command, followed by the next 255 bytes of the message, followed by a second one-byte flow command, and then the remaining 245 bytes of the message. The end of the packet would be followed by 10 synchronization bytes, and then another flow control command.

[0090] To distinguish data from commands, every eight bit byte is logically encoded in the network's switches using nine bits. The ninth bit is a flag indicating whether the byte is data or a command. As just explained, usually only one command is sent every 256 bytes. During normal operation of the network there two frequently used flow commands are: stop data flow and start data flow. During certain circumstances the normal flow of data is interrupted with other commands.

[0091] In the preferred embodiment, the nine-bit data/command values that are used in the switches are encoded for serial transmission by standard TAXI transmitter and receiver chips (model Am7968 and Am7969 integrated circuits made by Advanced Micro Devices (Trade Mark))

[0092] A third frequently used "command" is called a "synchronization byte". Synchronization bytes are simply null data and are considered to be "commands" in that they instruct the receiver that no data is being sent.

[0093] Figure 6 represents the signals being sent (TX) and received (RX) by host 132 over link 146. As noted above, each "link" between network members is actually two communications channels which simultaneously carry data in opposite directions. Thus Figure 6 shows two data streams. For example, referring to Figure 3, these data streams could represent the data streams on link 146 between host 132 and switch 142. For the purposes of this example, the TX data stream transmitted by host 132 contains a fairly long message packet, the beginning and end of which are denoted by a "B" byte and an "E" byte. The "B" byte represents the "begin packet command" that precedes every packet, and the "E" byte represents the "end packet command" that follows every packet.

[0094] "D" bytes represent the data in a message packet, and "0" bytes represent synchronization bytes which are sent when either there is no data to be transmitted or the flow of a packet has been temporarily halted.

[0095] The RX data stream sent by switch 142 to the host 132 contains flow control signals S (for start) and X (for stop) for controlling the flow of the packet being sent by the host 132. Stop commands sent by the switch 142 temporarily stop the flow of the packet being sent by the host, and start commands sent by the switch 142 cause the host 132 to resume sending the packet. The RX data stream sent by the switch 142 also contains a small data packet as denoted by the "B" and "E" bytes at the beginning and end of that packet.

[0096] As shown in Figure 6, a short time after the first start flow command 150 is sent by switch 142 (in the RX data

stream), the host begins to transmit its data packet. The host continues to transmit the data packet until a stop flow command X 152 is received. As will be explained in more detail below in the section entitled Switch Flow Control, the primary reason this might happen would be to prevent the data buffer in the receiving port of the switch from overflowing. When the switch is ready to receive more data it sends a start flow command S 154 and the host responds by resuming transmission of the data packet.

[0097] The flow control signal which is sent as every 256th byte is normally a "start flow" command, unless the packet buffer in the network member sending the command has less than a certain amount of free space left - which means that it is in danger of having a buffer overflow unless the flow of data is stopped. Thus, when no data is being received by a switch on a particular link, it continues to send "start flow" signals. It should be noted that each switch sends flow control signals at a particular time slot which is unrelated to the flow control time slots used by neighboring switches.

[0098] Host controllers 122 use the same flow control mechanism as the switches, except that host controllers 122 never send "stop flow" commands. Thus a host controller will always send "start flow" control signals to the switch that it is coupled to (i.e., every 256th byte). An example of this is shown in Figure 6 where the TX data stream contains "start flow" control signals 156, 158 and 160.

Host Network Controller

[0099] Next, we will describe in detail the primary hardware components of the network: the host controllers which couple host computers to the network, and the switches which handle the routing of data packets.

[0100] Referring to Figure 7, there is shown a block diagram of the network controller 122 for one host computer 120. Functionally, the network controller 122 is a port on the host computer for connecting the host to the network. This particular controller employs what is known as a Q22-Bus Control Protocol, using a Q-Bus control circuit 161 to couple the host computer to the controller 122. A description of the Q22-Bus protocol can be found in "Microsystems Handbook", published by Digital Equipment Corporation (1985). For host computers using other computer buses, different bus interface circuits would be used.

[0101] A microprocessor 162, an encryption circuit 164, and a error correction circuit 166 are used in normal fashion for encrypting messages and for generating error correction codes. All of these components of the controller 122 are coupled to a common data bus 168. Generally, the microprocessor 162 deposits a data packet received from the host 120 in the packet buffer 174 via the Q-Bus interface 161. The data packet from the host includes a command block that instructs the microprocessor 162 in the controller 122 on how to handle the packet. In particular, the controller 122 may be instructed to encrypt the packet using encryption circuit 164 with a specified encryption key. In addition, an error detection code is calculated using CRC circuit 166 and then appended to the end of the packet in the buffer 174.

[0102] Coupled to the data bus 168 are a data transmitting circuit 170 and a data receiving circuit 172. The data transmitting circuit 170 includes a packet buffer 174 that is used to store an entire packet before it is transmitted. The packet in the buffer 174 is transferred to a 1k byte FIFO (first in first out) buffer circuit 176 before being transmitted by transmitter 178 to a link interface circuit 180 via a link selector 182.

[0103] Link selector 182 selectively activates either link interface circuit 180 or link interface circuit 184. In the preferred embodiment, the link selector 182 under the control of the link control 186 automatically selects a predefined one of the two link interface circuits, such as circuit 180, unless the link coupled to that circuit is found not to be working (i.e., if no flow control commands are received on that link). If the normally selected link is not working, the link control circuit 186 causes the selector 182 to enable the other link interface circuit 184.

[0104] More specifically, the link control circuit 186 monitors the flow commands received by the receive path 172, and it detects the absence of flow control commands when flow commands are not received from the network on a regular basis. The circuit 186 informs the microprocessor 162 of the lack of flow control commands, and then the microprocessor 162 takes a number of steps to try to reestablish the controller's connection to the network. If these measures do not work, the microprocessor 162 sends a signal to the link control 186 to try the controller's other link interface circuit.

[0105] The following is a simplified explanation of how the flow control signals are used by each host controller 122 to select link interface 180 or 184 as the active link.

[0106] Upon power up, the host controller 122 begins transmitting synchronization signals on the initially selected link, and monitors the initially selected link for the receipt of flow control signals. If no flow control signals are received for a predefined period of time, the selector 182 is instructed to select the other available link. The process of looking for flow control signals on the currently selected link and switching links if none are detected continues until flow control signals are consistently received on one of the two links.

[0107] Link control circuit 186 monitors the flow control signals received by the currently active link interface 180 or 184. For the purposes of this initial explanation, it can be assumed that there are only two types of flow control signals: stop command signals and start command signals. When a start command signal is received, transmitter 178 is enabled and the data stored in packet buffer 174 is transmitted until either the packet buffer 174 is empty or until a stop command

signal is received. When a stop command signal is received, the link control circuit 186 "disables" the transmitter 178 so that synchronization signals (i.e., null data commands) are transmitted instead of new data from the packet buffer 174.

[0108] In the preferred embodiment, once the transmission of a packet is begun by the controller 122, the host controller 122 must always be ready to transmit all of the data in the packet on demand. In the preferred embodiment packets can be as small as about ten bytes, and as large as 16000 bytes.

[0109] Each complete packet that is to be transmitted is first stored in packet buffer 174 before transmission of the packet can begin. Then, link control circuit 186 enables the transmission of the packet in accordance with the flow control signals received on the active link, as described above.

[0110] The receive path includes a data receiver 190, a large (e.g., 4K byte) FIFO buffer 192, followed by a received packet buffer 194. As data packets are received from the active link, the data is initially stored in the FIFO buffer 192. From the FIFO buffer 192, which can hold many small packets, data is transferred into the packet buffer 194. When the end of a complete packet is detected, the packet in the buffer 194 is then processed (i.e., transferred to the host computer) and cleared from the packet buffer 194.

[0111] In the preferred embodiment the host controller 122 never sends out stop flow signals and must be prepared to receive a sequence of several packets. While one packet in the buffer 194 is being processed, other packets may be received and stored in the same buffer 194. Thus buffer 194 is a large dual ported circular buffer, with sufficient capacity (e.g., 128k bytes) for holding several large packets. Data is read out through one port of the buffer 194 for processing by the microprocessor 162 and transfer to the host 120, and new data packets are written via the other port of the buffer 194.

[0112] Using a large FIFO buffer 192 is generally preferred so that packets will not be lost due to slow processing by the host controller. If the FIFO buffer 192 does overflow, causing a packet to be lost, higher level protocols which require the acknowledgement of received packets cause the lost packets to be retransmitted.

[0113] The primary components of the link interface circuit 180 are two "TAXI" chips 196 and 198 (model Am7968 for the transmitter 196 and model Am7969 for the receiver 198, both integrated circuits made by Advanced Micro Devices) which are standard "transparent" asynchronous transmitter and receiver interface circuits. These circuits handle high speed data transmissions over point to point links, and thus are suitable for the 100 Megabit data transmission rates used in the preferred embodiment.

[0114] Detector 200 is a signal booster which helps the receiver circuit 198 handle weak input signals.

CUT-THROUGH, NONBLOCKING SWITCH

[0115] The switch 210, shown in Figures 8 and 9, is the key component of the entire network. The switch 210 is called a nonblocking switch because it can simultaneously interconnect several pairs of selected links. It is also called a cut-through switch because it can begin retransmitting (i.e., forwarding) data packets well before the complete packet has been received.

[0116] There is no central controller or intelligence which controls the network of the present invention. Rather, the network's intelligence and control logic, which makes routing decisions and handles various other network management tasks, is distributed over all of the switches in the network. For instance, each switch independently makes routing decisions, without knowledge as to the previous links used to transmit each data packet. However, the switches are designed so that each facilitates the efficient and error free routing of packets.

[0117] Referring first to the block diagram in Figure 8, the primary components of the switch 210 are a nonblocking crossbar switch 212, a number (twelve in the preferred embodiment) of switch ports 214 which are also called link control units 214, a switch control processor (SCP) 216, and a router 218 which is also called the routing logic circuit 218. There is also a special link circuit 214a for coupling the SCP 216 to the crossbar 212.

[0118] Each link unit 214 couples the crossbar 212 to one full duplex link 215. Each link 215 has two data channels so that data can be simultaneously transmitted in both directions over the link 215. Therefore each link unit 214 has two components: an input link unit 220 (Rx) and an output link unit 222 (Tx).

[0119] When a new data packet is received by the switch 210, the input link unit 220 which receives the data packet is coupled by the crossbar 212 to an output link unit 222A (for a different link than the input link). The output link unit 222 transmits the received data packet over another link, and thereby forwards the packet towards its destination.

[0120] As will be described in more detail with respect to Figure 9, the crossbar 212 is designed so that it can simultaneously couple any or all of the input link units 220 to distinct sets of output link units 222.

[0121] The purpose of the router 218 is to determine which output link unit 222 should be coupled to each input link unit 220. When a new data packet is received by an input link unit 220, the input link unit 220 sends a routing request to the router 218. The routing request specifies the destination of the packet, as well as the identity of the input link unit. As shown in Figure 8, the link unit 220 sends the packet's destination address to the router 218 over bus 230.

[0122] It is noted that the destination of the packet is stored in a few bytes at the beginning of each packet which

specify the network member to which the packet is being sent.

[0123] The Router Bus 232 includes a link mask with one bit corresponding to each of the link units, plus a four bit link index, a broadcast bit and a valid flag. Each of the lines of the link mask portion of bus 232 can be thought of as a single bit communication line between the router 218 and one of the link units 214.

[0124] An availability flag is periodically sent by each output link unit 222 to the router 218. The availability flag is ON when the output link is not busy and is "not blocked" and is therefore available for routing a new data packet. An output link unit is blocked when the switch on the other end of the link (i.e., the link coupled to the output link) unit has sent a Stop flow command. The Stop flow command indicates that the switch on the other side of the link is not ready to receive more data. When the output link unit 222 is busy or blocked, its availability mask is OFF. The thirteen availability mask bits from the output link units 222 are periodically sampled by the router 218 and then used to make a route selection.

[0125] Using the information sent by the input link unit 220, the router 218 determines which output link unit(s) 222 should be used to retransmit the data packet. The routing selection made by the router 218 is transmitted over the router bus 232 to the link units 214 and crossbar 212 which use the routing selection to set up the appropriate connections in the crossbar 212.

[0126] The router 218 is described below in more detail with respect to Figures 13-16. A preferred embodiment of the circuitry for the router 218 is described in US patent no. 5,179,558 entitled ROUTING APPARATUS AND METHOD FOR HIGH-SPEED MESH CONNECTED LOCAL AREA NETWORK. It is noted that while the initial preferred embodiment has only a dozen switch ports (i.e., link units) 214, it is anticipated that future units may have larger numbers of such ports.

[0127] The SCP 216 is a standard microprocessor (e.g., a 68010 microprocessor made by Motorola (Trade Mark) is used in the preferred embodiment) which is programmed to initialize the router 218 whenever the switch 210 is powered up or reset, and to perform a reconfiguration program whenever a component of the network fails or a new component is added to the network. The SCP is coupled to all the link units 214 by SCP bus 225 so that the SCP can monitor the status of the link units and can identify units which are not connected to a link and units which are malfunctioning.

[0128] Link unit 214a couples the switch control processor (SCP) 216 to the crossbar so that the SCP 216 can send and receive data packets via the crossbar 212 using the same communication mechanisms as the host computers in the network. During reconfiguration of the network, the SCP 216 sends data packets to the SCPs in the neighboring switches to determine the topology of the network, and to generate a new set of routing tables for the routers 212 in the network's switches.

[0129] Connections between input link units 220 and output link units are made by the crossbar 212 as follows. Generally, each time that the router 218 issues a new link selection, two multiplexers inside the crossbar will be set so that a selected input link unit is coupled to a selected output link unit. Two multiplexers are needed because one transmits data from the input link unit to the output link unit, while the other multiplexer transmits flow control signals back to the input link unit. When broadcast packets are transmitted, the number of multiplexers set up by the link selection signals will depend on the number of output links being used.

Crossbar Circuit

[0130] In Figure 9, the input and output portions 220 and 222 of each link unit have been separated so as to show their logical relationship to the crossbar 212. The input link units 220 are shown along the left side of the crossbar 212 while the output link units 222 are arrayed along the bottom of the crossbar 212. However, as will be explained below, the circuitry of these two units 220 and 222 is interconnected and the control logic for the two is not entirely separate. In addition, solely for the purposes of this one drawing, each input link unit 220 is shown a second time at the bottom of the crossbar 212 for reasons which will soon be explained.

[0131] As shown in Figure 9, each input link unit is coupled to a 9-bit wide data path 234 and 1-bit wide flow control line 236. The data path 234 carries data from data packets, and the flow control line 236 carries flow control information.

[0132] The crossbar 212 includes two multiplexers 240 and 242 for each link unit 214. The first multiplexer 240, called the data transmission multiplexer, couples a corresponding output link unit 222 to a selected one of the data paths 234. Since there are as many data transmission multiplexers 240 as there are link units 214, several or even all of the output link units 222 can be simultaneously coupled to corresponding selected ones of the input link units 220. In other words, the switch 212 is a nonblocking switch which can simultaneously route many packets.

[0133] In addition, it can be seen that two or more of the transmitting link units 222 can be coupled to the same data path 234 simply by causing their data transmission multiplexers 240 to select the same data path. This latter capability is used when broadcasting data packets to all the hosts on the network.

[0134] The second multiplexer 242, called the flow control multiplexer, couples a corresponding input link unit 220 to a selected one of the flow control lines 236. In other words, the flow control commands received by one input link

unit 220 are transmitted via the crossbar 212 to the control circuitry in another one of the input link units. Since there are as many flow control multiplexers 242 as there are link units 214, each input link unit 220 can be simultaneously coupled to a corresponding selected one of the other link units 214.

[0135] Each multiplexer 240 and 242 has an associated selection register (not shown) which is used to store a four-bit selection value that is sent to it by the router 218. These selection values determine which data path and flow control lines will be coupled to each of the link units.

[0136] In summary, the crossbar has one multiplexer 240 or 242 for directing data or flow commands to each of the input and output link units 220 and 222.

[0137] The selection signals for the multiplexers 240 and 242 are generated and output on router bus 232 by the router 218. Every time that the beginning of a new packet reaches the front of the FIFO buffer in an input link unit 220, the input link unit 220 transmits a routing request to the router 218 via bus line 230. The router responds to routing requests by generating and transmitting a multiplexer control signal over the router bus 232. The router bus 232 has the following components:

link mask
link index
broadcast flag
router bus valid flag.

[0138] Note that the operation of the router 218 and how it generates these values will be discussed below in the section entitled "Router Circuit".

[0139] The output link mask contains a separate ON/OFF flag for each of the output link units 222. Each output link 222 having a mask flag with a value of "1" will be coupled to a specified input link unit. The broadcast flag is set when a broadcast packet is being simultaneously routed to a plurality of network members. The router bus valid flag is set whenever the router 218 is asserting a route selection on the bus 232, and is reset otherwise.

[0140] The link mask portion of the router bus 232 is used to transmit bits corresponding to the selected output links, and the link index is a four-bit value that identifies the input link unit. The crossbar uses the four-bit link index as the multiplexer selection signal for the data transmission multiplexer(s) 240 coupled to the selected output link unit(s). For example, if the link mask has a "1" flag for output link unit 5 and the input link selection has a value of 0011 (i.e., 3), the value 0011 will be used as the selection signal for the multiplexer 240 associated with the fifth output link unit 222. If the output link mask has a "1" flag for several output link units, then the input link selection value will be used for each corresponding multiplexer.

[0141] The link index value that is transmitted by the router 218 is also used for setting up the flow control multiplexers 242. To do this, when the valid bit is ON, the crossbar circuit 212 remembers the link mask and link index which were sent by the router 218 and then sets up the flow control multiplexer 242 for the input link specified by the link index value. When the broadcast bit on the router bus is OFF, the selection value loaded in the flow control multiplexer 242 corresponds to the output link identified on the link mask portion of the bus.

[0142] When the data packet received by the input link unit 220 is being broadcast to more than one output link mask, the broadcast bit on the router bus is ON, and the selection value loaded into the flow control multiplexer 25 is a special value (e.g., 15). This causes the input link unit to use a special clock signal from a clock generator 246, called Clk256, in place of the normal flow control signals. As explained earlier, a broadcast packets are transmitted without regard to the normal flow control signals.

[0143] In summary, the router 218 transmits link selection values over bus 232 which is used by the crossbar circuit 212 to store corresponding values in the selection registers of the crossbar's multiplexers 240 and 242, and thereby causes the crossbar to couple the selected input and output link units.

[0144] The link selection values sent on the router bus 232 are also monitored by the input and output link units so as to coordinate the transmission of data packets through the crossbar 212 and then through the output link unit to another network member.

[0145] The operation of the router 218 and the signal protocols used on the router bus 232 are discussed in more detail below with reference to Figures 13-15.

Switch Flow Control

[0146] Referring to Figure 10, the basic mechanism for flow control between switches, and between switches and hosts is as follows. Every input link unit has an internal FIFO buffer which is used to temporarily store received data. Ideally, data should be read out of the FIFO buffer about as fast as it stored. But numerous factors, such as a blocked (i.e., busy) link, can cause data to back up in a FIFO buffer. When the FIFO buffer reaches a certain degree of fullness it sends out a stop flow command to the network member ("transmitter") that is sending it data. When the transmitter

receives the stop flow command, it temporarily stops sending data until a start flow command is received. The receiving FIFO buffer sends out start flow commands whenever enough data has been read out of the FIFO buffer so that it is less than a specified amount full.

[0147] Flow commands for any particular channel are transmitted in the opposite direction as the reverse channel of the same link, multiplexed with data that is transmitted on that reverse channel.

[0148] Of course, when a stop flow command is received by a switch, data will start to back up in a FIFO buffer in that switch, too. Thus there will be a chain reaction of stop flow commands that are generated as FIFO buffer's in each of the chain of switches becomes filled. Eventually, if the data packet is long enough, the host that is sending the packet may receive a stop flow command to temporarily stop it from sending out the rest of the packet.

[0149] There is also a chain reaction of start flow commands that is generated when the cause of the logjam goes away and the FIFO buffer which generated the first stop flow command is able to transmit the data that it has stored.

[0150] Figure 10 shows some of the details of the link units used in the preferred embodiment that are particularly relevant to the flow of data packets through the network. Figure 10 depicts the flow of a data packet from a host computer 120 through a sequence of two switches 300 and 302. When a data packet is received by link unit i in switch 300, it routes the packet by coupling the input link unit i to an output link unit TXj via its crossbar. The data packet then travels through a second switch 302, which routes the packet again. If the FIFO buffer 310, 340 in any of the Rx link units reaches half full, it forces the previous network member in the data transmission chain to stop transmitting data until the data already in the FIFO buffer can be processed.

[0151] As the host 120 begins to transmit a data packet over link 306, an input link unit 308 in switch 300 temporarily stores the received data in a FIFO buffer 310. Using the data in the packet's header (not shown), the switch 300 determines which of its links are proper links for forwarding the data packet towards its destination. In this example, link 312 is selected.

[0152] If link 312 is not busy, the switch connects the input link unit 308, through crossbar 320, to the output link unit 322 that is coupled to selected link 312. Usually, the switch 300 can select an output link and can connect the input link unit 308 to the output link unit 322 in less than the time that it takes to receive the first twenty-five bytes of a data packet.

[0153] However, if the link 312 is busy, the switch 300 will continue to store the data packet in the FIFO buffer 310 and will wait for an appropriate link to become available. In the preferred embodiment, the FIFO buffer 310 can store 4k bytes of packet data and has built-in circuitry which generates a half-full flag on line 324 when the buffer is at least half full. When the FIFO buffer 310 is less than half-full, this flag is interpreted as a "start flow" command; when the FIFO buffer is more than half-full, the flag is interpreted as a "stop flow" command.

[0154] The current value of the half-full flag from the FIFO buffer 310 is transmitted as a flow control value back to the host 120 by the output link unit 326 which is coupled to link 306. When the flow control value is "0", the data "throttle" 186 (i.e., the link control in Figure 7) in the host enables the transmission of data packets by the host 120. When the FIFO buffer 310 reaches half full, however, the "stop flow" command generated by the FIFO buffer causes the throttle 186 in the host computer 120 to temporarily stop the transmission of data by the host. When the switch 300 transmits enough data for the FIFO buffer 310 to become less than half full, the FIFO buffer 310 puts out a "start flow" command which enables the host's throttle 186 to resume transmitting the data packet.

[0155] As will be explained below with reference to Figure 11, there are some built in transmission delays and packet handling requirements which result in a requirement that the FIFO buffer 310 have about 2k bytes of room left in it when it first generates a "stop flow" command. In general, the minimum required size of the FIFO buffer 310 is a function of the maximum link length, and the maximum broadcast packet size. As a result, it has been found to be convenient to use a 4k x 9 FIFO buffer, such as the IDT 7204, which already includes circuitry that generates flags indicating whether the FIFO buffer is empty, and whether it is at least half full.

[0156] As shown in Figure 10, when the first switch 300 routes the data packet onto link 312, the data packet is received by another switch 302. There, it is once again stored in a FIFO buffer 340 inside an input link unit 342 while the switch decides where to route the packet. If there are no available links on which the data packet can be routed, the FIFO buffer 340 will generate a "stop flow" command on line 344 when it is half-full. This stop flow command is sent over link 312 to switch 1. In particular, it can be seen that the stop flow command is received by input unit 330 of switch 300, and that the flow command is then routed through the output link unit 332 and then the crossbar 320 to the input link unit 308 which is receiving the data packet. There, the flow command controls a throttle circuit 332, which enables the transmission of the data stored in FIFO buffer 310 when it receives a start flow command and disables transmission when it receives a stop flow command.

[0157] When link 346 becomes available, switch 302 begins transmitting the data in the FIFO buffer 340 over that link, via output link unit 348. When the FIFO buffer 340 becomes less than half-full, it sends out a start flow command over line 344, thereby enabling switch 300 to resume transmission of the data packet.

Link Units

[0158] Figure 11 provides a more detailed picture of the input and output link units of switch 300, previously shown in Figure 10. Each input link unit 308 and 330 includes a TAXI receiver chip 350 that converts the bit serial data received over an incoming link 306 or 312 into a 9-bit parallel signal that is transmitted over a 9-bit wide bus to a demultiplexer 352. Each byte of data contains a data type-flag, indicating whether the byte is data or a command, which comprises the ninth bit of each byte.

[0159] The demultiplexer 352 monitors the type-flag of the signals received from the TAXI Rx circuit 350, and splits off commands in the data stream from data. Data signals, as well as end of packet "command bytes", are stored in the FIFO buffer 310. Flow control commands received over the link 306 are converted into an ON/OFF (i.e., 1/0) binary signal which is transmitted on line 354. The flow control command on line 354 is latched in a latch 356 that is clocked with the transmission clock Clk256 of the corresponding output link unit 322. The latched flow control signal is then ANDed by AND gate 358 with the transmission clock CLk256, and the resulting signal is sent through the crossbar 320 for transmission to another input link unit 308. The output of the AND gate 358 is coupled by the crossbar 320 to throttle control line 360 in input link unit 308.

[0160] The latch 356 and AND gate 358 cause the flow control signals sent to the input link unit 308 to be synchronized with the transmission clock of the output link unit 322. In addition, the AND gate 358 causes the transmitted flow command to be OFF once every 256 bytes so as to stop the transmission of data through the crossbar 320 for one byte, during which time the output link unit 322 transmits a flow control signal instead of data. In essence, the output link unit 322 puts out a "stop flow" command on throttle control line 360 every 256th byte cycle, as determined by clock Clk256, so that the throttle 322 of the corresponding FIFO buffer 310 will not send data during the flow control cycle of the switch.

[0161] Thus, as described above, flow control signals received by an input link unit are latched and synchronized by the corresponding output link unit, and are then used to start and stop the flow of data through that output link unit.

[0162] Each output link unit 322 converts the 9-bit parallel signals received from the crossbar 320 into bit serial signals that are transmitted over an output link 312. More specifically, the output link unit 322 contains a multiplexer 362. The multiplexer 362 is coupled to clock Clk256, which alternately enables the transmission of data from line 364 for 255 data byte cycles, and then enables the transmission of one flow command byte. A clock with the same period as Clk256 is coupled to the demultiplexer 352 so that the FIFO buffer 310 does not, on average, fill faster than it can be emptied.

[0163] The multiplexer 362 derives the flow commands that it sends from the status of line 366. Line 366 carries the half-full flag generated by the FIFO buffer 310 in the input link unit 330. Generally, when the FIFO buffer 310 is at least half-full, an ON (i.e., STOP) signal will be sent on line 366, and otherwise an OFF (i.e., START) signal will be sent on line 366. The signal on 366 is converted by an encoder circuit 368 into a nine-bit "stop flow" or "start flow" command for transmission by the Taxi Tx circuit 370.

[0164] The data and flow commands output by the multiplexer 362 are converted into a bit-serial data stream by TAXI transmitter 370, which transmits the multiplexed data and commands over link 312.

[0165] Figure 12 shows additional details of the link unit circuitry. The demultiplexer 352 in input link unit 308 as shown in Figure 11 is shown in Figure 12 to be implemented using a pipeline register 380, status registers 382, and control logic 384. All received data is stored for one byte cycle in the pipeline register 380, which gives the control logic 384 time to determine whether each byte should be loaded into the FIFO buffer 310. Flow commands are decoded and stored in the status registers 382. The control logic 384 receives a clock signal on line 385 that is synchronized with the data being received. This clock signal is generated by The Taxi Rx circuit 350. The control logic 384 reads the status registers 382 and disables the loading of data into the FIFO buffer 310 when certain commands are received. More generally, the control logic 384 is a finite state machine which generates a set of clocks signals that used to control the flow of data through the part of the input link unit up to and including the input port of the FIFO buffer 310.

[0166] It should be noted that the input side of the FIFO buffer 310 is clocked by signals synchronized with the data being received by TAXI Rx circuit 350, while the output side of the FIFO buffer 310 is clocked by a different clock signal generated by an independent clock circuit in the switch. The two clock rates are approximately equal, within about 0.02%, but are not synchronized.

[0167] With the assistance of a sequence of pipeline register 390 at the output of the FIFO buffer 310, a second control logic circuit 392 identifies the beginning of each new packet, which contains the packet's destination address. The packet's destination address is sent to the router via buffer 394.

[0168] The throttle 332 shown in Figure 11 is implemented by the control logic 392 which generates the output clock signals for the FIFO buffer 310 and pipeline register 390. The control logic 392 receives flow control signals from line 354. Note that the received flow control signals were transmitted through the crossbar by another input link unit. When a stop flow command is received, the control logic 392 simply disables the output clock signal for the FIFO buffer 310 and pipeline register 390, thereby halting the flow of data out of the FIFO buffer 310.

[0169] The control logic 392 also monitors the data/command bit of each 9-bit byte of data as it is read out of the FIFO buffer 310 so as to identify the end of each packet. Only data and end of packet command bytes are stored in the FIFO buffer 310. Therefore the end of a packet is detected by the control logic 392 when an enabled command bit is read from the FIFO buffer 310. After the end of each packet, the control logic 392 waits until the current packet has cleared the pipeline, and then begins looking for a new data packet to be forwarded.

[0170] The control logic 392 interacts with the router 318 via the router bus 232. When the beginning of a new packet is detected, the control logic 392 sends a routing request signal on the link mask portion of the router bus 232 and receives a "grant" signal on the same link mask portion of the router bus during a later time slot. When a grant signal is received, the packet destination address for the new packet is asserted by buffer 392 on bus 230. The control logic 392 also synchronizes the transmission of a new data packet with routing selection signals sent by the router on bus 232.

[0171] Both logic circuits 384 and 392 store status signals in the status registers 382 indicating the current status of the input link unit 308. The switch control processor (SCP) periodically reads some of the status values stored in the status registers 382 to determine which link units are coupled to a live link and which link units are working properly.

[0172] The output link unit 326, as shown in Figure 12, consists of a pipeline register 402, a decoder 404, a finite state machine (FSM) 406, and a TAXI transmitter 370. Data from the crossbar is held for one clock cycle in the pipeline register 402 to allow setup of decoder 404, as required by the TAXI timing specifications. Whenever an end of packet command byte is received in the pipeline register 402, the FSM 406 recognizes that command and changes its internal state. Thereafter, if the corresponding output link is not blocked by STOP flow control signals received by the input link unit 308, the FSM 406 then sends out a "link available" signal to the router 218 so that the router will know that this link is available for routing a new packet. The FSM 406 also commands the TAXI Rx circuit 370 to send out an end of packet command byte and then commands the TAXI 370 to transmit synchronization bytes until the router 218 reconnects the output link 326 to an input link for transmitting another packet.

[0173] The decoder 404, in conjunction with the FSM 406, acts as the multiplexer 362 of Figure 11. In particular, the FSM 406 uses the Clk256 clock signal to determine when the TAXI transmits data from the crossbar and when it transmits flow commands. The decoder 404 receives the FIFO half-full status signal from the input link unit. During each time period for transmitting a flow control signal, the decoder 404 decodes the FIFO half-full signal so as to form an appropriate command for the TAXI 370. At the beginning of each packet it forms a BEGIN command and at the end of each packet the decoder 404 forms an END command. If the output link unit is blocked by a STOP flow command, or if the output link unit is idle, the decoder 404 forms a SYNC command. During all other time periods, the decoder 404 sends a "data transmission" command to the TAXI 370. The FSM 406 determines the status of the output link unit 326 and what command the decoder 404 should send to the TAXI 370.

[0174] The output link FSM 406 also synchronizes the transmission of a new data packet with routing selection signals sent by the router on bus 232. The same routing selection signals are used by the route selection logic 408 in the crossbar to set up the data and flow multiplexers for coupling a specified input link unit to one or more specified output link units.

[0175] Flow Control for Broadcast Packets. As will be described below in the section on broadcast packets, the transmission of broadcast packets cannot be stopped in the middle of a packet. Since there is a predetermined maximum size for broadcast packets (e.g., 1528 bytes), there must be room in the FIFO buffer 310 (shown in Figure 11) to absorb an entire broadcast packet that is just starting to be sent when a stop flow command is generated by the switch receiving the packet.

[0176] To determine the amount of room which must be left in the FIFO buffer 310 when it sends out a stop flow command in order to be able to receive a complete broadcast packet, the following factors must be included: the maximum delay before the stop flow command is sent, the maximum amount of data that may have already been transmitted when the transmitting network member receives and acts on the stop command, and the maximum size of a broadcast packet. The maximum delay before the stop flow command may be sent is 256 bytes. In addition, for a 2 kilometer fiber optic cable with a 100 megabits/sec transmission rate, the amount of data that may have already been transmitted when the transmitting network member receives the stop command is about 260 bytes. Adding the above delay factors, the FIFO buffer 310 needs at least 2044 (256 + 260 + 1528) bytes of unused storage when it generates a stop flow command so that it can absorb a broadcast packet that it is about to be sent without losing any of the data in the packet. To account for miscellaneous delays and to provide an additional safety margin, the FIFO buffer 310 generates a stop flow command when it has 2k (i.e., 2048) bytes of storage left.

[0177] In the preferred embodiment, each input FIFO buffer 310 is large enough to store 4k bytes of data. These FIFO buffers 310 are designed to generate start flow commands as long as they are less than half full (i.e., with more than 2k bytes left unused) and to generate a stop command when they are at least half full (i.e., with 2k bytes or less unused).

[0178] Packet Continuity. A packet or data underrun occurs when a switch that has partially transmitted a message is ready to receive more of the message from a previous switch, but the previous switch is not ready to transmit the rest of the message. Several aspects of the present invention include features which make packet underrun impossible.

In terms of the circuitry of Figure 10, these features are designed so as to guarantee that until the end of a packet is received, there will always be data in the FIFO buffer 310 to be transmitted.

[0179] First, the host controller of Figure 7 is programmed so that the transmission of a packet is not started until the entire packet is stored in packet buffer 174. This ensures that the host controller can transmit the remainder of a packet upon request.

[0180] Second, referring now to Figure 8, whenever a switch receives a new data packet, it takes a period of time for the router 218 to process the routing request for that data packet and to determine which output link should be used to retransmit the packet. During the time that the router 218 is working on this, at least twenty-five bytes of data are stored in the FIFO buffer 310.

[0181] The only remaining requirement to prevent packet underrun is that it must be impossible to read all the data in the FIFO buffer 310 before more data reaches it from the previous network member. Basically, this means that there is a limit on the amount of the clock rate mismatch for the data transmitters in each of the switches. For instance, if the transmitter in a switch is slightly faster than the transmitter in the previous network member, the amount of data in the FIFO buffer 310 will slowly decrease as the packet traverses the switch. Therefore, to prevent packet underrun the maximum amount of time that it takes to transmit the largest legal data packet multiplied by the maximum clock rate discrepancy must be less than the amount of data stored in the FIFO buffer 310 before the transmission of new packet is enabled.

[0182] In the preferred embodiment, the maximum length packet is 16k bytes long, and the maximum clock skew is about 0.02 per cent. As a result, the amount of data which initially needs to be stored in the FIFO buffer 310 to prevent packet underrun is approximately 4 bytes. In the preferred embodiment, it takes the router 218 at least twenty-five byte cycles (at 100 Megabits/second) to make a route selection, and it takes the switch at least one more byte cycle to couple the input link unit to the selected output link unit. Thus at least twenty-five bytes will be stored in the FIFO buffer 310 before the retransmission of the packet can begin.

[0183] It is noted that one benefit of the flow control system used by the present invention is that it avoids the need for the control logic 392 in the input link units to examine the FIFO buffer 310 to detect packet underruns, and therefore avoids the need to synchronize the output side of the FIFO buffer 310 with the input side. While synchronized FIFO access circuits are available and would solve any clock rate mismatches between switches, such circuits are much more expensive than the buffering scheme of the present invention.

ROUTER CIRCUIT

[0184] Every switch in the network is assigned a unique seven-bit SHORT ID in addition to its 48-bit UID. SHORT IDs are assigned during configuration of the network and the SHORT ID for any particular switch may change when the network is reconfigured. Each host computer is assigned an eleven-bit "network address". The network address of a host computer is generated by concatenating the SHORT ID of its switch with the four-bit value of the link port which couples the host to the switch. The network address of each switch is its SHORT ID plus a predefined four-bit value (e.g., zero) corresponding the link number of the SCP link unit.

[0185] Network addresses are the address values used to specify the destinations of packets transmitted through the network.

[0186] The reason that each network member is assigned a network address as well as a UID is that a shorter value was needed to facilitate the routing of packets through the network. The seven-bit SHORT ID allows for up to 128 switches. Since each switch has at most twelve external ports, at least one of which must be used to connect the switch to another switch in the network, there can be at most 1408 hosts. This is expected to be more than sufficient for all anticipated applications of the present invention. Of course, the allowed number of network members could be doubled simply by using a 12-bit network address.

[0187] When a data packet is first transmitted, the network address of the network member to which the data packet is being sent is stored in the first few bytes of the packet. The router 218 uses the value of the short address, as well as the input link on which the packet is received, to determine which output link(s) should be used to retransmit a data packet.

[0188] Generally, the purpose of the router 218 is to allocate system resources (i.e., output links) on a fair and equitable basis to data packets. It is also the job of the router 218 to prevent packet starvation. The router uses a first come, first considered routing priority wherein requests for resources are compared with the set of available resources in the order that the requests were received. The first request to match the available resources is selected and allocated the resources that it needs. Then the process repeats.

[0189] Using the first come, first considered routing discipline, later requests can be allocated resources before an earlier request as long as the allocation doesn't conflict with the needs of the earlier request. This routing discipline maximizes the rate at which available resources can be allocated to resource requesters. For broadcast data packets, this routing discipline means that requested resources are reserved by broadcast requests, thereby preventing later

requests from impeding the progress of broadcast data packets.

[0190] Figure 13 shows the basic components of the router circuit 218 used in the preferred embodiment. As was shown in Figure 9, the router 218 receives packet destination addresses on bus 230. Routing requests and output link availability signals are time-multiplexed on router bus 232 along with the transmission of link selection values by the router 218.

[0191] Each "routing address" includes an eleven-bit packet address and a four-bit input link number. The routing address is stored in a register 420. A routing table 422 is a look up table which is indexed by routing address values. The routing table 422 contains an entry, for every possible routing address value, which specifies the output links which could potentially be used for routing the packet that corresponds to the routing address.

[0192] Whenever an input link unit detects the receipt of a new packet at the output of its FIFO buffer, it sends a request signal on the link mask portion 232A of the router bus 232.

[0193] A routing request selector circuit 424 monitors bus 232A to see if any routing requests are being asserted. If one or more routing requests are asserted during any one routing engine cycle, the selector 424 selects one of the requests. The selected request is acknowledged by sending an ON signal on bus 232A to the selected link unit at an appropriate time. This acknowledgment signal instructs the signaled link unit that it has been selected to transmit its routing request over bus 230, and then the selected input link unit sends the packet destination address for its routing request to buffer 420 via bus 230.

[0194] The request selector circuit 424 is a cyclic priority encoder, which bases the priority for selecting among competing requests on the last link unit whose request was selected. This ensures that all requests are accepted within a short period of time and helps to prevent packet starvation.

[0195] Each routing table address includes an eleven-bit packet destination address received on line 230, and its associated four-bit input link number, which is provided by the request selector circuit 424. The routing table address is stored in a register 420 for use by a routing table 422. The routing table 422 is stored in a random access memory and the fifteen bit value in register 420 is used as the address for retrieving a value (called a routing mask) from the routing table 422. The selected routing mask output by the routing table 422 is latched in by the routing engine 430 at the beginning of the next routing engine cycle, as will be explained in more detail below.

[0196] Figure 14 shows how a packet address, sent on line 230, is derived from the first two data bytes of an arriving packet, and how that data is combined with the input link number generated by request selection circuit 424. See also Figure 12. Note that in the preferred embodiment, the packet address is fifteen bits long. In future embodiments, the number of bits used for the packet address or the input link number may be increased.

[0197] Routing table 422 contains an entry 426 for every possible routing address. In other words, it has an entry for every possible combination of a 4-bit input link number with an 11-bit packet address. Since these two values occupy fifteen bits, the number of entries in the table 422 will be 2^{15}, or 32,768. Each entry occupies two bytes of storage, and therefore the table 422 requires 65,536 bytes of storage. Typically, only a small number of the entries in the routing table will represent "legal" routing requests, and all the others will represent corrupted or otherwise illegal request values. The table entry for illegal requests is BC=1, with the remaining portion of the mask equal to all zeros. If a data packet generates an illegal routing request, the data packet is purged from the switch.

[0198] It may be noted that the reason that the routing table 42 is indexed by input link number and network address, rather than being indexed only by network address is as follows. If the network addresses in packets were never corrupted, the routing table could be indexed by network address. The entries in the routing table would still follow the up/down routing rule. This is possible because from any given position in the network there will always be at least one path to a specified network address which will not violate the up/down routing rule, assuming that the packet traveled on a legal route to its current position in the network. That legal path can be stored in switch routing tables that are indexed only by network address. However, if a network address in a packet were corrupted, and routing tables were not indexed by input link number, it would possible to have deadlock. This is because a packet could "take a wrong turn" after its destination network address was corrupted.

[0199] In a network with eighty network members there will be a total of only eighty-one or so legal packet addresses, including one address for each network member and one or more "broadcast" addresses for sending packets to all hosts on the network. Also, some combinations of input link numbers and packet addresses will be illegal because they correspond to routes which take packets away from their destination or create possible deadlocks. Therefore, in a eighty member network the routing table 422 for any particular switch would be likely contain between 320 and 750 legal entries.

[0200] Each entry in the routing table 422 contains a link vector, which is also called a routing mask. An example of a routing mask entry is:

ADDRESS VALUE		ROUTING MASK	
Input Link,	Packet Address	BC |	0123456789AB (Link#)
0110	11001100110	0	001110000000

[0201] Each address in the routing table represents one of the possible routing request values that can be received from an input link, and is therefore represented here by the concatenation of an input link number and a packet address.

[0202] The routing mask in each routing table entry 426 contains thirteen mask bits, one for each of the output links of the switch including the SCP. Each mask bit which is ON (i.e., equal to "1") represents an output link which may be used to route the packet. The routing mask also contains a broadcast bit BC which indicates whether the packet address is a broadcast address or a normal address. An example of a routing mask 425 is shown in Figure 14, which also shows a valid bit above the routing mask and a link number below it for reasons that are explained below.

[0203] If the broadcast bit BC is ON (i.e., equal to "1"), the packet is called a broadcast packet. Broadcast packets must be simultaneously forwarded to all of the output links specified by the routing mask.

[0204] If the broadcast bit is OFF (i.e., equal to "0"), the packet is called a non-broadcast packet. For a non-broadcast packet the routing mask has a mask bit equal to "1" for each output link which could be used to route the packet toward its destination (i.e., the packet may be routed on any single one of the output links specified by the routing mask). In many instances, several different alternate output links can be used to route a packet toward its destination, which is one of the advantages of mesh connected networks. The routing engine 430 selects just one of the output links specified by the routing mask for routing the packet.

[0205] The bit values in the routing mask of each routing table entry 426 are determined by the up/down routing rule, discussed above. In accordance with the up/down routing rule, the set of legal routes for a data packet depends on whether the last link used (i.e., the link used to get to the current switch) was an up link or a down link. If the previous switch transmitted the packet on a down link, only down links may be used by the next switch. However, if the previous switch used an up link, both up and down links may be legally used by the next switch. In addition, the set of usable links denoted in each routing mask only includes those links which will move the data packet closer to its destination.

[0206] Figure 14 shows the format of a "routing request" 428 as it is read into the routing engine 430. The top bit, called the valid flag is set to "1" whenever a routing request is being loaded into the routing engine, and is reset to "0" when no new routing requests are being processed. The next fourteen bits are the link vector obtained from the selected entry of the routing table 422, as discussed above. The last four bits are the input link number for the packet being routed.

[0207] Routing engine 430 compares a link availability mask, which represents the currently available output links, with routing requests. More particularly, the purpose of the routine engine 430 is to match the output link needs of each new packet with the available output links of the switch. The routing selection value generated by the routing engine 430 is used by the crossbar 212 (shown, for example, in Figures 8 and 9) to set up its multiplexers and thereby connect a specified input link to one or more specified output links. The routine engine is the subject of US patent no. 5,179,558 entitled ROUTING APPARATUS AND METHOD FOR HIGH-SPEED MESH CONNECTED LOCAL AREA NETWORK,

[0208] As described with respect to Figure 12, each output link unit 326 transmits a "link available" signal which indicates whether that output link is available for routing, or is already either in use or blocked. Bus 232 carries the link available signal lines from all the output links. The routing engine 430 samples the link available signals on bus 232 at the beginning of each new routing engine cycle. The routing engine 430 then uses the available link mask for making routing selections.

[0209] When the routing engine 430 is able to match a routing request with one or more available links, it generates a routing selection value which it outputs on bus 232. The routing selection value consists of the four bit input link number, the broadcast bit and the valid bit from the satisfied routing request, and an output link mask which identifies the output link or links that are to be coupled to the input link. The input link number, the broadcast bit and the valid bit are transmitted on the portion of the router bus labelled 232B, and the output link mask is transmitted on the portion of the router bus labelled 232A. The route selection values transmitted on router bus 232 are used by the input and output link units 220 and 222, and crossbar 212 (shown in Figure 9) to connect a specified input link to one or more specified output links.

[0210] The "valid" output bit is ON only in cycles during which the routing engine 430 outputs a new route selection. Thus the "valid" bit output by the routing engine 430 is OFF in cycles during which the routing engine 430 is unable to match any of the pending routing requests with the available output links.

[0211] Control circuit 435 generates clock signals for the routing engine 430 and request selection circuit 424. These clock signals also control the use of the packet address bus 230 and the router bus 232. That timing protocol will be described below with reference to Figure 16.

[0212] The control logic 435 is also used by the SCP 216 to reload the routing table 428 during reconfiguration of

the network, to keep track of the status of the router 218, and to load certain firmware in the routing engine 430 upon power up or resetting of the entire switch.

Routing Engine

[0213] Figure 15 shows a preferred embodiment of the routing engine 430. In this embodiment, the routing engine is formed from an array 450 of computational components, each of which is represented by a box in Figure 15. The array shown represents a programmable gate array called the Xilinx 3090 array, made by Xilinx Corp (Trade Mark). The Xilinx 3090 contains sixteen columns with twenty combinational logic blocks (CLBs) in each column. The CLBs can be electrically programmed to perform a variety of logic and storage functions. Each CLB contains two flip-flops and two function units. Each function unit is capable of calculating any boolean function of up to four input variables. The CLB produces two outputs, which can come directly from the function blocks or from the flip flops. There are also two tri-state drivers near each CLB. These drivers can be connected to horizontal metal traces that cross the chip, allowing the construction of buses. In addition to providing programmable logic, the Xilinx 3090 array provides programmable interconnections between neighboring CLBs, as well as a number of pad cells which provide an interface to circuits outside the array. Thus, the behavior and function of the array is determined by a pattern of control bits which is loaded into the array from an external source (e.g., the SCP in each switch). No customization is done as part of chip manufacturing.

[0214] The routing engine array 450 uses thirteen columns 451 - 463, each with nineteen logic blocks. Each of these columns 451-463 stores and processes a single routing request. In addition, on the right side of the array there is a column 465 of thirteen ready signal generators (RG) and a column 468 of thirteen output signal generators (O).

[0215] Routing requests are received on the left side of the array. The signal symbols shown of the left side of the array match the format of the routing request shown in Figure 14.

[0216] An output link availability mask is received on the right side of the array 450. The output link availability mask is represented by signals RDY0 through RDY12, and is received from buffer 440 as shown in Figure 13.

[0217] Outputs from the array 450, which are the routing selections made by the routing engine, emerge on bus 470 from the right side of the array. As described above with reference to Figure 13, the routing selection contains nineteen bits: a valid bit, indicating a routing selection has been made, a thirteen bit output mask, and the broadcast bit and the four bit input link number from the routing request.

[0218] The thirteen columns 451-463 of the array act as a queue which implements the first come, first considered routing discipline of the router. The columns at the right side of the queue hold the oldest unsatisfied routing requests, while those on the left hold more recent requests.

[0219] The entire array works on a periodic clock cycle. The routing engine accepts one routing request per clock cycle and makes one attempt to make a routing selection during each clock cycle.

[0220] Referring to Figure 16, each router clock cycle has six phases labelled T0 through T5. Each phase lasts 80 nanoseconds, for a total router clock cycle of 480 nanoseconds. The router clock cycle has two major subphases represented by clock signal T03. During the first subphase T03=1 and during the second subphase T03=0.

[0221] As will now be described, it takes three router cycles to send a routing request to the router 218, to process the request, and then to send a routing selection to the link units and crossbar.

[0222] Routing requests are sent to the router 218 as follows. During T4, each input link unit which has a routing request that needs to be sent to the router asserts a ON signal on its corresponding line of router bus 232. The routing selection circuit 424 monitors the router bus 232 during T4 to see if any routing requests are being asserted. If only one request is asserted, it is acknowledged. If more than one routing request is asserted during any one clock cycle, the routing selection circuit 424 selects just one of the requests, as was described above.

[0223] The selected request is acknowledged by sending an ON signal on bus 232 to the selected link unit during T3 of the next router cycle. This acknowledgment signal instructs the signaled link unit that it has been selected to transmit its routing request over bus 230. During clock phases T3 through T5 the selected input link unit sends the packet address for its routing request to the routing table 422 via buffer 420. During phases T3 through T5 the routing table 422 is accessed and the link vector corresponding to the routing request is ready at its output by the end of T5.

[0224] During phase T5 all the output link units assert their availability flag values on the router bus 232 so that these signals will be ready for the routing engine at the beginning of the next router cycle.

[0225] At the beginning of T0, the routing engine 430 latches in the link availability flags from router bus 232 and the current routing request, if any. The current routing request comprises the link vector output by the routing table 422, and the link number and valid bit output by the request selection circuit 424.

[0226] During the rest of the router cycle, T0 through T5, the routing engine 430 compares the latched in link availability data with all the unsatisfied routing requests stored in the data columns of the routing engine 430. The result of that comparison is latched in the output column 468 of the routing engine at the end of T5. However, the routing selection generated by the routing engine is not asserted on the router bus 232 until T1 of the following router cycle. During T1

through T5 of this router cycle, if the Valid bit of the routing selection is ON, the link units and crossbar process the routing selection output so as to couple the specified input link unit with the specified output link unit(s). The link units also prepare to begin transmitting the data in the specified input link unit's FIFO 310.

[0227] During T3 of this router cycles the crossbar circuit 212, which remembers the input link number asserted by the routing engine and the specified output link(s), asserts an output link number on the link index portion of the router bus for setting up the flow control multiplexer corresponding to the specified input link number. If the broadcast bit in the routing selection is ON, however, the output link number asserted during T3 is set to a predefined number (e.g., 15 or F).

[0228] In summary, each portion of the router 218 performs a distinct task during each six part router cycle. In addition, the router bus 232 is time multiplexed for sending routing requests to the routing request selector 424 and for sending routing selections to the link units.

[0229] Used in a three stage pipeline with six 80ns clock cycles per stage, the router 218 can route slightly more than two million packets per second, and adds a latency of about 1.44 microseconds per switch in the path of each packet. The three stages of the router pipeline are (1) input link selection and routing table lookup to generate a routing request mask, (2) the routing engine cycle, and (3) transmission of routing selections to the crossbar 212 and the link units.

[0230] The following is a more detailed description of the operation of the routing engine during each phase of the router cycle. At the beginning of each router cycle, at the beginning of T0, a routing request and the available output link mask are read in. The routing request is latched into the leftmost column of the array 451, and the link availability mask (RDY0 to RDY12) is latched into the ready signal column 465. In addition, each unsatisfied routing request which is already stored in the array is shifted one column to the right in the array if there is at least one column to its right in the array which is not occupied by an unsatisfied request.

[0231] During the first subphase of the router cycle several sets of signals propagate across the array. First, the link availability mask propagates across the array from right to left. The circuitry in each of the request handling columns 451-463 compares the routing request stored in that column with the link availability mask. In those columns which store non-broadcast requests (with BC=0) a MATCH signal is generated if at least one enabled MASK bit matches an enabled RDY bit.

[0232] In those columns which store broadcast requests (with BC=1), a MATCH signal is generated only if all of the enabled MASK bits match the corresponding RDY bits (i.e., only if all output links needed by the request are available).

[0233] Columns which store broadcast requests (with BC=1) also block the propagation of those RDY signals which match the MASK bits of the broadcast request. In effect, broadcast requests "reserve" the available output links needed by that request. if this were not done, the routing of a broadcast packet could be permanently stymied by subsequent requests which match and use individual ones of the output links needed by the broadcast packet.

[0234] The MATCH signals are propagated upwards through those columns where a match is found. Thus the MATCH signals are the second set of signals which propagate during the first phase of the clock cycle.

[0235] It is quite possible for two or more columns to generate MATCH signals. In order to give the oldest unsatisfied requests first consideration it is necessary to select the rightmost column in which a match was found. To do this a signal called ANSWERED propagates through the C1 cells at the top of the array from the right side of the array to the left. The ANSWERED signal has a value of "0" until it encounters a valid column (i.e., VALID="1") with an enabled MATCH signal, at which point ANSWERED is set equal to "1".

[0236] The ANSWERED signal is the third type of signal which propagates during the first subphase of the router cycle.

[0237] At the end of the T3, an output enable signal ND_ENABLE is generated for the rightmost column with an enabled MATCH signal that receives an ANSWERED signal of "0" from its right-hand neighbor. Of course, during many clock cycles none of the columns will match the available link mask, and no output enable signal will be generated. For the moment, consider the case in which an output enable signal is generated for one selected column.

[0238] Only one column, at most, will have an enabled ND_ENABLE signal during any one clock cycle. If none of the columns have an enabled ND_ENABLE signal, that means that the routing engine failed to find any routing requests which matched the available output links.

[0239] During the second subphase of the router cycle, the following data values from the column with the enabled ND_ENABLE signal are propagated to the output column 468 of the array: all the matched routing mask bits (i.e., enabled mask bits for which there is an available output link), the broadcast bit, link number bits and the valid bit.

[0240] The circuitry in the output column 468 works as follows. For non-broadcast requests (BC=0), only the lowest of the enabled mask bits is output, and all the other mask bits are disabled. For broadcast requests (BC=1), all the enabled mask bits are output. For both types of request, the broadcast bit, link number bits and valid bit are also output by the output column 468.

[0241] The resulting routing selection value will have an enabled valid bit, one enabled mask bit (unless it is a broadcast packet), and the broadcast bit and input link number of the routing request.

[0242] It will be clear to those who consider the matter that some packets cannot be routed immediately because sometimes the output link or links that a packet needs will be busy. Therefore columns 451-463 of the routing engine array 450 act as a queue in which unsatisfied routing requests are stored and periodically compared with the available output links.

[0243] When the routing engine fails to match the available output links with any pending routing requests, the data output by the array has a disabled VALID bit. The link units and crossbar circuit in the switch ignore outputs of the routing engine during cycles in which the VALID bit is disabled.

[0244] As new requests are entered into the array 450, request handling columns containing unsatisfied or invalid requests must shift to the right to make room for the new routing requests. The data values stored in all the request handling columns 451-463 in the array will be shifted to the right during each cycle, except for columns at the right side of the array which hold unsatisfied requests.

[0245] More specifically, each column 451-463 loads in the data from the column to its left (and the leftmost column loads from the routing request buffer 428) if certain conditions are met:

(1) if the column contains an invalid request, the column will be overwritten with the data from the column to the left, or
(2) at least one column to the right will load data from the column to its left, or
(3) the routing request in the column has been satisfied and selected for output.

[0246] If a column supplies the result and no column to its right shifts, the request will be overwritten by the contents of the column to its left. If, on the other hand, the array contains an invalid request to the right of a column that supplies a result, then the already-satisfied request will shift right by one column and will remain in the array.

[0247] To ensure that such a request does not supply a result in a subsequent cycle, the inputs of the registers which store the BC and VALID values in each column are ANDed with the ND_ENABLE signal from the column to the left. This means that if a column supplies a result and shifts in the same cycle, the request will be invalidated and its BC bit cleared when it shifts into the next column to the right.

[0248] The final outputs generated by the output column 468 are stored in latches (not shown) in the output column 468 at the end of each cycle. The output signals stored in the output column are transmitted on the router bus 232 during the T1 subcycle of the next routing engine cycle.

[0249] The RDY signals supplied by the output link monitoring subsystem (i.e., via latching buffer 440 in Figure 13) cannot be used directly by the routing engine. This is because the router 218 is part of a pipelined circuit. If the routing engine 430 supplies an output that uses a particular output link, then that output link must be made to appear not ready (i.e., not available) for use during the following clock cycle of the router. This is accomplished by ANDing the incoming RDY mask with the complement of the output mask before delivering it to the queue of routing requests in columns 451-463 of the routing engine.

Broadcast Packet Handling

[0250] Broadcast packets are generally messages sent by one host computer to all the other hosts in the network. While other types of broadcast packets are possible, including broadcast packets initiated by a switch and limited distribution broadcasts, the same routing and starvation prevention considerations apply to all types of broadcast packets.

[0251] Broadcast packets are typically the most difficult type of packets to handle in mesh connected networks because of the need to simultaneously transmit broadcast messages over many network links. It is also necessary to ensure that broadcast packets proceed quickly to all destinations because of the important functions served by broadcast packets.

[0252] A typical use of a broadcast packet is as follows. Host A wants to send a message to Host B. However, Host A does not know Host B's network address - i.e., the eleven-bit address that is stored in the first two bytes of every packet. Therefore Host A needs a mechanism for obtaining the network address of Host B. This is a well known problem in local area networks with a well known solution. The solution is to send a "broadcast message" or packet to all the hosts in the system. The content of the broadcast message is: "Host B, please send a message containing your network address to Host A at network address X". The broadcast message is sent to every host computer in the network, but only Host B (if there is one) will respond by sending the requested information to Host A. Then Host A can send its message to Host B. In a well designed local area network, this entire transaction will typically take a very small fraction of a second.

[0253] Another time that broadcast messages are used is when Host A tries to send a packet to Host B, but Host B does not acknowledge receipt of the packet. A typical protocol for this situation is for Host A to resend the packet. Then if no acknowledgment is received, Host A concludes that it does not have the correct address for Host B (e.g., because

the address has become corrupted, or because, unbeknownst to Host A, the address of Host B has been changed). Then Host A uses the broadcast packet protocol just described to find out Host B's current network address.

[0254] While many of the situations in which broadcast packets are useful are well known, it is not well known how to route a broadcast packet in a mesh connected network. The following is how it is done in the present invention.

[0255] Referring to Figure 17, there is shown a schematic representation of a mesh connected network 700. The network has sixteen switches 710 - 740, and twenty-seven hosts H1 - H27. Switch 710 is called the root node of the network, for reasons that will be explained below. This Figure is different from Figure 2 primarily in that the links between switches have two characteristics assigned to them: (1) a primary direction known as "up", and (2) so-called "spanning tree" links are shown with darker lines than "additional" links. The switches 710-740 and the spanning tree links collectively form a "spanning tree".

[0256] Consider the path followed by a broadcast packet initiated by host H19. The packet is given a unique eleven-bit "broadcast packet address", such as "8FF". First the packet travels up the network tree to the root, through switches 732, 724, 712 and then 710. To do this, the routing table in each of these switches will have a link vector entry corresponding to the input link used and the broadcast packet address. The link vector in each of the switches 732, 724 and 712 will indicate that the only route available to the packet is along links which move the packet up toward the root of the tree 710.

[0257] When the packet reaches the root switch 710, the broadcast packet will be rerouted down all the spanning tree links (shown in Figure 17 with bolder lines than the additional links) to all the hosts in the network. The routing table in the root has the same broadcast link vector value for the broadcast packet, regardless of the input link on which it arrives. In particular, the broadcast link vector specifies that received broadcast packets are simultaneously transmitted over all the spanning tree links coupled to the switch. Note that this is the only case in which a received data packet may be retransmitted over the same link as it was received on.

[0258] In the network 700 shown in Figure 17, the broadcast packet will be simultaneously transmitted from the root switch 710 to switches 712, 714, 716, 718 and 720. Also at the same time, the broadcast packet will be sent to any hosts coupled to the root switch (although none are shown in Figure 17). This transmission will occur when all of the required links become available.

[0259] At each of the receiving switches, the broadcast packet will be retransmitted to (1) all hosts coupled the switch, and (2) to all neighboring switches which are coupled to the transmitting switch by a down spanning tree link. In other words, the broadcast packet is sent down the spanning tree until every host in the network has received the broadcast packet.

[0260] In summary, broadcast packets are first routed up the network's spanning tree to the root, and then are transmitted down the tree to all the hosts in the network.

[0261] While the present invention has been described with reference to a few specific embodiments, the description is illustrative of the invention and is not to be construed as limiting the invention. Various modifications may occur to those skilled in the art without departing from the true scope of the invention as defined by the appended claims.

Claims

1. A mesh connected local area network for interconnecting a multiplicity of hosts, said network comprising:

a multiplicity of switch means (124, 126, 140, 142, 210) for simultaneously routing a multiplicity of data packets between hosts (120, 132, 134, 136, 138) in the network, each switch means (210) including a multiplicity of port means (214) for coupling the switch means to other switch means and hosts, and a non blocking crossbar switch (212) for simultaneously coupling and routing data packets between a multiplicity of selected pairs of said port means (214); said hosts and switch means together comprising network members; and
a multiplicity of point to point link means (128) for interconnecting said switch means and the hosts in said network, each point to point link means (128) providing a communication channel between two of said network members;
each said port means (214) including buffer means (310, 340) for buffering data packets received by said switch means at said port means (214) and cut-through means (392) for beginning to retransmit a received data packet through said non blocking crossbar switch before the end of said received data packet has been received;

characterized by:

said multiplicity of link means (128) including spanning tree links and a multiplicity of additional links;
said multiplicity of switch means (124, 126, 140, 142) and said spanning tree links together comprising a

spanning tree in which one of said switch means is designated the root (710) of said spanning tree;
each said data packet having a specified host to which said data packet is being sent;
each said switch means (124, 126, 140, 142) including routing means (218) for defining legal data packet transmission routes through said network; said routing means denoting a first subset of said link means as up links and denoting all other ones of said link means as down links;
said routing means defining for each received data packet a legal subset of the link means (128) coupled to said switch means through which said received data packet is allowed to be retransmitted, wherein said legal subset for data packets received on any of said up links is selected from said up links and down links coupled to said switch means and said legal subset for data packets received on any of said down links is selected only from said down links coupled to said switch means, so that the path by which each said data packet is transmitted through said network is composed of zero or more up links followed by zero or more down links;
whereby said routing means provides deadlock free routing of data packets through said mesh connected local area network.

2. The mesh connected local area network of claim 1, wherein:

said link means (128) are arranged in pairs, each pair of link means including one link means denoted as an up link for transmitting data packets from a first respective one of said network members to a second respective one of said network members and one link means denoted as a down link for transmitting data packets from said second respective one of said network members to said first respective one of said network members;
each port means (214) of each said switch means is coupled to two link means, including one up link and one down link, that couple said switch means to another network member; and
said spanning tree is defined such that said second respective one of said network members is defined to be closer to said root (710) of said spanning tree than said first respective one of said network members.

3. The mesh connected local area network of claim 1 or 2, further characterized by:

said switch means including link status means (382, 406) for generating link availability signals denoting which of said link means (128) is available for retransmitting a data packet;
said routing means including route selection means (430) for comparing said legal subset of said link means (128) through which a received data packet can be retransmitted with said link availability signals and for routing said received data packet through a link means that is included in said legal subset of said link means and that is available for retransmitting a data packet.

4. The mesh connected local area network of claim 3, said route selection means (430) including means (435) for periodically comparing said legal subset of said link means (128) through which a received data packet can be retransmitted with said link availability signals until one of the link means denoted as available matches a link means included in said legal subset of said link means, and then routing said received data packet through the matching link means.

5. The mesh connected local area network of claim 1, 2, 3 or 4, further characterized by:

said data packets including data packets that are sent to a single specified one of said hosts in said network and broadcast data packets that are to be sent to all said hosts in said network;
said routing means including means for denoting two broadcast packet transmission routes, including means for designating as an uplink port one port means (214) of said switch means that couples said switch means to another switch means that is closer to said root of said spanning tree, in accordance with predefined criteria, and for denoting as downlink ports each port means (214) that couples with spanning tree links said switch means to another switch means that is further from said root of said spanning tree, in accordance with said predefined criteria;
said routing means of each said switch means, except said switch means designated as said root (710) of said spanning tree, routing broadcast data packets received by said uplink port to all of said downlink ports, and for routing to said uplink port broadcast data packets received by any of said port means (214) other than said uplink port.

6. The mesh connected local area network of claim 6, further characterized by:
said routing means of said switch means designated as said root of said spanning tree including means for routing all broadcast data packets received by said switch means to all of said downlink ports.

7. The mesh connected local area network of claim 5 or 6,

said broadcast data packets having a predefined maximum size;
said buffer means (310, 34) having more than sufficient room to store an entire broadcast data packet of said predefined maximum size and means for indicating when said buffer means (310, 340) has sufficient room to receive an entire broadcast data packet;
said port means (214) including flow control means (406) coupled to said buffer means (310, 340) for sending flow control signals to a network member coupled to said port means (214), said flow control signals including stop flow signals requiring said network member to stop transmitting data packets to said port means (214) and start flow signals allowing said network member to resume sending data packets to said port means (214);
said flow control means sending start flow signals only when said buffer means (310, 340) has sufficient room to receive an entire broadcast data packet.

8. The mesh connected local area network of claim 7, said port means (214) including means (392) for receiving said flow control signals sent by a network member coupled to said port means (214) and for stopping the transmission of a data packet when a stop flow signal is received from said network member, unless said data packet is a broadcast packet.

9. A method of operating a mesh connected local area network, the network including

a multiplicity of switch means (124, 126, 140, 142, 210) for simultaneously routing a multiplicity of data packets between hosts (120, 132, 134, 136, 138) in the network, each switch means (210) including a multiplicity of port means (214) for coupling the switch means to other switch means and hosts, and a nonblocking crossbar switch (212) for simultaneously coupling and routing data packets between a multiplicity of selected pairs of said port means (214); said hosts and switch means together comprising network members; and
a multiplicity of point to point link means (128) for interconnecting said switch means and the hosts in said network, each point to point link means (128) providing a communication channel between two of said network members;
each said port means (214) including buffer means (310, 340) for buffering data packets received by said switch means at said port means (214) and cut-through means (392) for beginning to retransmit a received data packet through said nonblocking crossbar switch before the end of said received data packet has been received;

said method comprising the steps of:

at each said switch means, upon receiving a beginning portion of a data packet, when predefined link availability criteria are satisfied, beginning retransmission of said received data packet through said nonblocking crossbar switch before the end of said received data packet has been received, and otherwise buffering said received data packet until said predefined link availability criteria are satisfied;

characterized by:

configuring said network members as a spanning tree in which one of said switch means is designated the root (710) of said spanning tree and in which said multiplicity of link means (128) includes spanning tree links that are part of said spanning tree and a multiplicity of additional links that are not a part of said spanning tree;
defining legal data packet transmission routes through said network, including denoting a first subset of said link means as up links and denoting all other ones of said link means as down links;
at each said switch means, defining for each received data packet a legal subset of the link means (128) coupled to said switch means through which said received data packet is allowed to be retransmitted, wherein said legal subset for data packets received on any of said up links is selected from said up links and down links coupled to said switch means and said legal subset for data packets received on any of said down links is selected only from said down links coupled to said switch means, and routing each received data packet through a link means included in said defined legal subset of link means, so that the path by which each said data packet is transmitted through said network is composed of zero or more up links followed by zero or more down links;
whereby said defining and routing steps provide deadlock free routing of data packets through said mesh connected local area network.

10. The method of claim 9, further including the steps of :

arranging said link means (128) in pairs, each pair of link means including one link means denoted as an up link for transmitting data packets from a first respective one of said network members to a second respective one of said network members and one link means denoted as a down link for transmitting data packets from said second respective one of said network members to said first respective one of said network members;
coupling each port means (214) of each said switch means to two link means, including one up link and one down link, that couple said switch means to another network member; and
defining said spanning tree such that said second respective one of said network members is defined to be closer to said root (710) of said spanning tree than said first respective one of said network members.

11. The method of claim 9 or 10, further characterized by:

at each switch means (382, 406), generating link availability signals denoting which of said link means (128) is available for retransmitting a data packet;
said routing step including comparing said legal subset of said link means through which a received data packet can be retransmitted with said link availability signals and routing said received data packet through a link means that is included in said legal subset of said link means and that is available for retransmitting a data packet.

12. The method of claim 11, further characterized by:
said routing step including periodically comparing said legal subset of said link means (128) through which a received data packet can be retransmitted with said link availability signals until one of the link means denoted as available matches a link means included in said legal subset of said link means, and then routing said received data packet through the matching link means.

13. The method of claim 9, 10, 11 or 12, further characterized by:

said data packets including data packets that are sent to a single specified one of said hosts in said network and broadcast data packets that are to be sent to all said hosts in said network;
said step of defining legal data packet transmission routes including denoting two broadcast packet transmission routes, including designating as an uplink port one port means (214) of said switch means that couples said switch means to another switch means that is closer to said root (710) of said spanning tree, in accordance with predefined criteria, and for denoting as downlink ports each port means (214) that couples with spanning tree links said switch means to another switch means that is further from said root (710) of said spanning tree, in accordance with said predefined criteria;
said routing step as performed by each said switch means, except said switch means designated as said root (710) of said spanning tree, including routing broadcast data packets received by said uplink port to all of said downlink ports, and routing to said uplink port broadcast data packets received by any of said port means (214) other than said uplink port.

14. The method of claim 13, further characterized by :
said routing step, as performed by said switch means designated as said root (710) of said spanning tree, including routing all broadcast data packets received by said switch means to all of said downlink ports.

15. The method of claim 12 or 13, further characterized by:

said broadcast data packets having a predefined maximum size;
providing each said port means (214) with a buffer means (310,34) having more than sufficient room to store an entire broadcast data packet of said predefined maximum size and means for indicating when said buffer means (310, 34) has sufficient room to receive an entire broadcast data packet;
at each said port means (214), sending flow control signals to a network member coupled to said port means (214), said flow control signals including stop flow signals requiring said network member to stop transmitting data packets to said port means (214) and start flow signals allowing said network member to resume sending data packets to said port means (214); said sending step sending start flow signals only when said buffer means (310, 34) has sufficient room to receive an entire broadcast data packet.

16. The method of claim 15, further characterized by:

at each said port means (214), receiving said flow control signals sent by a network member coupled to said port means (214), and stopping the transmission of a data packet when a stop flow signal is received from said network member, unless said data packet is a broadcast packet.

Patentansprüche

1. Vermaschtes lokales Netz zum Verbinden mehrerer Hosts, wobei das Netz enthält:

mehrere Schalteinrichtungen (124, 126, 140, 142, 210) zum gleichzeitigen Lenken mehrerer Datenpakete zwischen Hosts (120, 132, 134, 136, 138) im Netz, wobei jede Schalteinrichtung (210) mehrere Anschlußeinrichtungen (214) zum Koppeln der Schalteinrichtung mit anderen Schalteinrichtungen und Hosts enthält sowie einen nicht sperrenden Crossbarschalter (212) zum gleichzeitigen Verbinden mehrerer ausgewählter Paare von Anschlußeinrichtungen (214) und zum gleichzeitigen Lenken von Datenpaketen zwischen diesen Anschlußeinrichtungen (214); wobei die Hosts und die Schalteinrichtungen zusammen Netzelemente enthalten; und
mehrere Punkt-zu-Punkt-Verbindungseinrichtungen (128) zum Verbinden der Schalteinrichtungen und der Hosts im Netz, wobei jede Punkt-zu-Punkt-Verbindungseinrichtung (128) zwischen zwei der Netzelemente einen Kommunikationskanal schafft;
wobei jede der Anschlußeinrichtungen (214) Puffereinrichtungen (310, 340) zum Puffern von Datenpaketen, die von den Schalteinrichtungen an den Anschlußeinrichtungen (214) empfangen werden, sowie Durchschalteinrichtungen (392) zum Beginnen einer Rücksendung eines empfangenen Datenpakets durch den nicht sperrenden Crossbarschalter, bevor das Ende des empfangenen Datenpakets empfangen worden ist, enthält;

dadurch gekennzeichnet, daß:

die mehreren Verbindungseinrichtungen (128) Überbrückungsbaum-Verbindungen sowie mehrere weitere Verbindungen enthalten;
die mehreren Schalteinrichtungen (124, 126, 140, 142) und die Überbrückungsbaum-Verbindungen zusammen einen Überbrückungsbaum enthalten, worin eine der Schalteinrichtungen als Wurzel (710) des Überbrückungsbaums bestimmt ist;
jedes Datenpaket an einen spezifischen Host geschickt wird;
jede Schalteinrichtung (124, 126, 140, 142) eine Lenkungseinrichtung (218) zum Definieren zulässiger Datenpaketübertragungswege durch das Netz enthält; wobei die Lenkungseinrichtung eine erste Untermenge der Verbindungseinrichtungen als Aufwärtsverbindungen und sämtliche anderen der Verbindungseinrichtungen als Abwärtsverbindungen be- bzw. kennzeichnet;
die Lenkungseinrichtungen für jedes empfangene Datenpaket eine zulässige Untermenge der Verbindungseinrichtungen (128), die mit den Schalteinrichtungen gekoppelt sind, über die das empfangene Datenpaket zurückgesendet werden kann, definieren, wobei die zulässige Untermenge für Datenpakete, die von irgendeiner der Aufwärtsverbindungen empfangen werden, aus den mit den Schalteinrichtungen gekoppelten Aufwärtsverbindungen und Abwärtsverbindungen gewählt wird und die zulässige Untermenge für Datenpakete, die über irgendeine der Abwärtsverbindungen empfangen werden, nur aus den Abwärtsverbindungen gewählt wird, die mit den Schalteinrichtungen gekoppelt sind, so daß der Weg, über den jedes Datenpaket durch das Netz geschickt wird, aus null oder mehr Aufwärtsverbindungen, gefolgt von null oder mehr Abwärtsbverbindungen, gebildet ist;
wobei die Lenkungseinrichtungen eine Lenkung der Datenpakete durch das vermaschte lokale Netz ohne gegenseitige Blockierung schaffen.

2. Vermaschtes lokales Netz nach Anspruch 1, bei dem:

die Verbindungseinrichtungen (128) paarweise angeordnet sind, wobei jedes Paar Verbindungseinrichtungen eine als Aufwärtsverbindung gekennzeichnete Verbindungseinrichtung zum Senden von Datenpaketen von einem jeweils ersten der Netzelemente an ein jeweils zweites der Netzelemente und eine als Abwärtsverbindung gekennzeichnete Verbindungseinrichtung zum Senden von Datenpaketen vom jeweils zweiten der Netzelemente an das jeweils erste der Netzelemente enthält;
jede Anschlußeinrichtung (214) jeder Schalteinrichtung mit zwei Verbindungseinrichtungen gekoppelt ist, die eine Aufwärtsverbindung und eine Abwärtsverbindung umfassen und die Schalteinrichtung mit einem weiteren Netzelement koppeln; und

der Überbrückungsbaum in der Weise definiert ist, daß das jeweils zweite der Netzelemente als näher als das jeweils erste der Netzelemente an der Wurzel (710) des Überbrückungsbaums befindlich definiert ist.

3. Vermaschtes lokales Netz nach Anspruch 1 oder 2, ferner dadurch gekennzeichnet, daß

die Schalteinrichtungen Verbindungsstatuseinrichtungen (382, 406) zum Erzeugen von Verbindungsverfügbarkeitssignalen, die angeben, welche der Verbindungseinrichtungen (128) zum Zurücksenden eines Datenpakets verfügbar ist, enthalten;
die Lenkungseinrichtungen Wegauswahleinrichtungen (430) zum Vergleichen der zulässigen Untermenge der Verbindungseinrichtungen (128), über die ein empfangenes Datenpaket zurückgesendet werden kann, mit den Verbindungsverfügbarkeitssignalen sowie zum Lenken des empfangenen Datenpakets über eine Verbindungseinrichtung, die in der zulässigen Untermenge von Verbindungseinrichtungen enthalten ist und zum Zurücksenden eines Datenpakets verfügbar ist, enthalten.

4. Vermaschtes lokales Netz nach Anspruch 3, bei dem die Wegauswahleinrichtungen (430) Einrichtungen (435) zum periodischen Vergleichen der zulässigen Untermenge von Verbindungseinrichtungen (128), über die ein empfangenes Datenpaket zurückgesendet werden kann, mit den Verbindungsverfügbarkeitssignalen, bis eine der als verfügbar bezeichneten Verbindungseinrichtungen mit einer Verbindungseinrichtung übereinstimmt, die in der zulässigen Untermenge von Verbindungseinrichtungen enthalten ist, und dann zum Lenken des empfangenen Datenpakets über die übereinstimmende Verbindungseinrichtung enthalten.

5. Vermaschtes lokales Netz nach Anspruch 1, 2, 3 oder 4, ferner dadurch gekennzeichnet, daß:

die Datenpakete diejenigen Datenpakete, die an einen einzelnen spezifizierten der Hosts im Netz gesendet werden, sowie Rundsende-Datenpakete, die an sämtliche Hosts im Netz geschickt werden sollen, umfassen;
die Lenkungseinrichtungen Einrichtungen zum Kennzeichnen zweier Rundsendepaket-Übertragungswege enthalten, die ihrerseits Einrichtungen, die eine Anschlußeinrichtung (214) der Schalteinrichtung, die diese Schalteinrichtung mit einer weiteren Schalteinrichtung koppelt, die sich näher an der Wurzel des Überbrükkungsbaums befindet, in Übereinstimmung mit im voraus definierten Kriterien als Aufwärtsverbindungsanschluß kennzeichnen und jede Anschlußeinrichtung (214), die mit Überbrückungsbaumverbindungen die Schalteinrichtung mit einer weiteren Schalteinrichtung koppelt, die sich weiter entfernt von der Wurzel des Überbrückungsbaums befindet, in Übereinstimmung mit den im voraus definierten Kriterien als Abwärtsverbindungsanschlüsse kennzeichnen, enthalten;
die Lenkungseinrichtung jeder Schalteinrichtung mit Ausnahme derjenigen Schalteinrichtung, die als Wurzel (710) des Überbrückungsbaums bestimmt ist, die Rundsende-Datenpakete, die vom Aufwärtsverbindungsanschluß empfangen werden, an sämtliche der Abwärtsverbindungsanschlüsse lenkt und an die Aufwärtsverbindungsanschlüsse Rundsende-Datenpakete lenkt, die von irgendeiner der Anschlußeinrichtungen (214), die von dem Aufwärtsverbindungsanschluß verschieden sind, empfangen werden.

6. Vermaschtes lokales Netz nach Anspruch 5, ferner dadurch gekennzeichnet, daß:
die Lenkungseinrichtung der als Wurzel des Überbrückungsbaums bestimmten Schalteinrichtung eine Einrichtung enthält zum Lenken sämtlicher Rundsende-Datenpakete, die von der Schalteinrichtung empfangen werden, an sämtliche Abwärtsverbindungsanschlüsse.

7. Vermaschtes lokales Netz nach Anspruch 5 oder 6,

wobei die Rundsende-Datenpakete eine im voraus definierte maximale Größe besitzen;
die Puffereinrichtungen (310, 34) mehr als ausreichenden Raum zum Speichern eines vollständigen Rundsende-Datenpakets mit der im voraus definierten maximalen Größe und außerdem eine Einrichtung besitzen, die angibt, wenn die Puffereinrichtungen (310), 340) ausreichenden Raum besitzt, um ein vollständiges Rundsende-Datenpaket zu empfangen;
wobei die Anschlußeinrichtungen (214) Flußsteuereinrichtungen (406) enthalten, die mit den Puffereinrichtungen (310, 340) gekoppelt sind, um Flußsteuersignale an ein mit der jeweiligen Anschlußeinrichtung (214) gekoppeltes Netzelement zu schicken, wobei die Flußsteuersignale Flußanhaltesignale, die das Netzelement auffordern, die Übertragung von Datenpaketen an die Anschlußeinrichtung (214) anzuhalten, sowie Flußstartsignale, die dem Netzelement erlauben, das Schicken von Datenpaketen an die Anschlußeinrichtung (214) wiederaufzunehmen, enthalten; wobei die Flußsteuereinrichtung Flußstartsignale nur dann schickt, wenn die Puffereinrichtung (310, 340) ausreichenden Raum für den Empfang eines vollständigen Rundsende-Daten-

pakets besitzt.

8. Vermaschtes lokales Netz nach Anspruch 7, wobei die Anschlußeinrichtungen (214) Einrichtungen (392) enthalten zum Empfangen der Flußsteuersignale, die von einem mit der jeweiligen Anschlußeinrichtung (214) gekoppelten Netzelement geschickt werden, sowie zum Anhalten der Übertragung des Datenpakets, wenn ein Flußanhaltesignal vom Netzelement empfangen wird, sofern das Datenpaket kein Rundsendepaket ist.

9. Verfahren zum Betreiben eines vermaschten lokalen Netzes, wobei das Netz enthält:

mehrere Schalteinrichtungen (124, 126, 140, 142, 210) zum gleichzeitigen Lenken mehrerer Datenpakete zwischen Hosts (120, 132, 134, 136, 138) im Netz, wobei jede Schalteinrichtung (210) mehrere Anschlußeinrichtungen (214) zum Koppeln der Schalteinrichtung mit anderen Schalteinrichtungen und Hosts enthält sowie einen nicht sperrenden Crossbarschalter (212) zum gleichzeitigen Verbinden mehrerer ausgewählter Paare von Anschlußeinrichtungen (214) und zum gleichzeitigen Lenken von Datenpaketen zwischen diesen Anschlußeinrichtungen (214); wobei die Hosts und die Schalteinrichtungen zusammen Netzelemente enthalten; und

mehrere Punkt-zu-Punkt-Verbindungseinrichtungen (128) zum Verbinden der Schalteinrichtungen und der Hosts im Netz, wobei jede Punkt-zu-Punkt-Verbindungseinrichtung (128) zwischen zwei der Netzelemente einen Kommunikationskanal schafft;

wobei jede der Anschlußeinrichtungen (214) Puffereinrichtungen (310, 340) zum Puffern von Datenpaketen, die von den Schalteinrichtungen an den Anschlußeinrichtungen (214) empfangen werden, sowie Durchschalteinrichtungen (392) zum Beginnen einer Rücksendung eines empfangenen Datenpakets durch den nicht sperrenden Crossbarschalter, bevor das Ende des empfangenen Datenpakets empfangen worden ist, enthält;

wobei das Verfahren die folgenden Schritte enthält:

bei jeder Schalteinrichtung bei Empfang eines Anfangsabschnitts eines Datenpakets dann, wenn im voraus definierte Verbindungsverfügbarkeitskriterien erfüllt sind, Beginnen einer Zurücksendung des empfangenen Datenpakets durch den nicht sperrenden Crossbarschalter, bevor das Ende des empfangenen Datenpakets empfangen worden ist, und andernfalls

Puffern des empfangenen Datenpakets, bis die im voraus definierten Verbindungsverfügbarkeitskriterien erfüllt sind;

gekennzeichnet durch:

Konfigurieren der Netzelemente als einen Überbrückungsbaum, in dem eine der Schalteinrichtungen als die Wurzel (710) des Überbrückungsbaums bestimmt ist und in dem die mehreren Verbindungseinrichtungen (128) Überbrückungsbaum-Verbindungen, die Teil des Überbrükkungsbaums sind, sowie mehrere zusätzliche Verbindungen, die nicht Teil des Überbrückungsbaums sind, enthalten;

Definieren zulässiger Datenpaket-Übertragungswege über das Netz, einschließlich des Kennzeichnens einer ersten Untermenge der Verbindungseinrichtungen als Aufwärtsverbindungen und des Kennzeichnens sämtlicher anderen Verbindungseinrichtungen als Abwärtsverbindungen;

bei jeder Schalteinrichtung für jedes empfangene Datenpaket Definieren einer zulässigen Untermenge der mit der Schalteinrichtung gekoppelten Verbindungseinrichtungen (128), über die das empfangene Datenpaket zurückgesendet werden darf, wobei die zulässige Untermenge für über irgendeine der Aufwärtsverbindungen empfangene Datenpakete aus den Aufwärtsverbindungen und Abwärtsverbindungen, die mit der Schalteinrichtung gekoppelt sind, gewählt wird und die zulässige Untermenge für über irgendeine der Abwärtsverbindungen empfangene Datenpakete nur aus den mit der Schalteinrichtung gekoppelten Abwärtsverbindungen gewählt wird, und Lenken jedes empfangenen Datenpakets über eine in der definierten zulässigen Untermenge von Verbindungseinrichtungen enthaltene Verbindungseinrichtung, so daß der Weg, über den jedes der Datenpakete durch das Netz übertragen wird, aus null oder mehr Aufwärtsverbindungen, gefolgt von null oder mehr Abwärtsverbindungen, gebildet ist;

wobei die Definitions- und Lenkungsschritte eine Lenkung von Datenpaketen durch das vermaschte lokale Netz ohne gegenseitige Blockierung schaffen.

10. Verfahren nach Anspruch 9, ferner mit den folgenden Schritten:

paarweises Anordnen der Verbindungseinrichtungen (128), wobei jedes Paar Verbindungseinrichtungen eine als Aufwärtsverbindung gekennzeichnete Verbindungseinrichtung zum Senden von Datenpaketen von einem jeweils ersten der Netzelemente an ein jeweils zweites der Netzelemente und eine als Abwärtsverbindung

gekennzeichnete Verbindungseinrichtung zum Senden von Datenpaketen vom jeweils zweiten der Netzelemente an das jeweils erste der Netzelemente enthält;
Koppeln jeder Anschlußeinrichtung (214) jeder Schalteinrichtung mit zwei Verbindungseinrichtungen, die eine Aufwärtsverbindung und eine Abwärtsverbindung umfassen und die Schalteinrichtung mit einem weiteren Netzelement koppeln; und
Definieren des Überbrückungsbaums in der Weise, daß das jeweils zweite der Netzelemente als näher als das jeweils erste der Netzelemente an der Wurzel (710) des Überbrückungsbaums befindlich definiert ist.

11. Verfahren nach Anspruch 9 oder 10, ferner dadurch gekennzeichnet, daß:

bei jeder Schalteinrichtung (382, 406) Erzeugen von Verbindungsverfügbarkeitssignalen, die angeben, welche der Verbindungseinrichtungen (128) zum Zurücksenden eines Datenpakets verfügbar ist;
der Lenkungsschritt den Vergleich der zulässigen Untermenge der Verbindungseinrichtungen (128), über die ein empfangenes Datenpaket zurückgesendet werden kann, mit den Verbindungsverfügbarkeitssignalen sowie das Lenken des empfangenen Datenpakets über eine Verbindungseinrichtung, die in der zulässigen Untermenge von Verbindungseinrichtungen enthalten ist und zum Zurücksenden eines Datenpakets verfügbar ist, enthält.

12. Verfahren nach Anspruch 11, ferner dadurch gekennzeichnet, daß:
der Lenkungsschritt das periodische Vergleichen der zulässigen Untermenge von Verbindungseinrichtungen (128), über die ein empfangenes Datenpaket zurückgesendet werden kann, mit den Verbindungsverfügbarkeitssignalen, bis eine der als verfügbar bezeichneten Verbindungseinrichtungen mit einer Verbindungseinrichtung übereinstimmt, die in der zulässigen Untermenge von Verbindungseinrichtungen enthalten ist, und dann das Lenken des empfangenen Datenpakets über die übereinstimmende Verbindungseinrichtung enthalten.

13. Verfahren nach Anspruch 9, 10, 11 oder 12, ferner dadurch gekennzeichnet, daß

die Datenpakete diejenigen Datenpakete, die an einen einzelnen spezifizierten der Hosts im Netz gesendet werden, sowie Rundsende-Datenpakete, die an sämtliche Hosts im Netz geschickt werden sollen, umfassen;
der Schritt des Definierens zulässiger Datenpaket-Übertragungswege das Angeben zweier Rundsendepaket-Übertragungswege enthält, das seinerseits das Bezeichnen einer Anschlußeinrichtung (214) der Schalteinrichtung, die diese Schalteinrichtung mit einer weiteren Schalteinrichtung koppelt, die sich näher an der Wurzel des Überbrückungsbaums befindet, in Übereinstimmung mit im voraus definierten Kriterien als Aufwärtsverbindungsanschluß und jeder Anschlußeinrichtung (214), die mit Überbrückungsbaumverbindungen die Schalteinrichtung mit einer weiteren Schalteinrichtung koppelt, die sich weiter entfernt von der Wurzel des Überbrückungsbaums befindet, in Übereinstimmung mit den im voraus definierten Kriterien als Abwärtsverbindungsanschlüsse enthält;
der Lenkungsschritt, der von jeder der Schalteinrichtungen mit Ausnahme derjenigen Schalteinrichtung, die als Wurzel (710) des Überbrückungsbaums bestimmt ist, ausgeführt wird, die Rundsende-Datenpakete, die vom Aufwärtsverbindungsanschluß empfangen werden, an sämtliche der Abwärtsverbindungsanschlüsse lenkt und an die Aufwärtsverbindungsanschlüsse Rundsende-Datenpakete lenkt, die von irgendeiner der Anschlußeinrichtungen (214), die von dem Aufwärtsverbindungsanschluß verschieden sind, empfangen werden.

14. Verfahren nach Anspruch 13, ferner dadurch gekennzeichnet, daß
der Lenkungsschritt, der von der als Wurzel des Überbrückungsbaums bestimmten Schalteinrichtung ausgeführt wird, das Lenken sämtlicher Rundsende-Datenpakete, die von der Schalteinrichtung empfangen werden, an sämtliche Abwärtsverbindungsanschlüsse enthält.

15. Verfahren nach Anspruch 12 oder 13, ferner dadurch gekennzeichnet, daß

die Rundsende-Datenpakete eine im voraus definierte maximale Größe besitzen;
jede der Anschlußeinrichtungen (214) mit einer Puffereinrichtung (310, 34) versehen wird, die mehr als ausreichenden Raum zum Speichern eines vollständigen Rundsende-Datenpakets mit der im voraus definierten maximalen Größe und außerdem eine Einrichtung besitzt, die angibt, wenn die Puffereinrichtungen (310), 340) ausreichenden Raum besitzt, um ein vollständiges Rundsende-Datenpaket zu empfangen;
wobei bei jeder der Anschlußeinrichtungen (214) Flußsteuereinrichtungen (406) gesendet werden, die mit den Puffereinrichtungen (310, 340) gekoppelt sind, um Flußsteuersignale an ein mit der jeweiligen Anschlußeinrichtung (214) gekoppeltes Netzelement zu schicken, wobei die Flußsteuersignale Flußanhaltesignale, die

das Netzelement auffordern, die Übertragung von Datenpaketen an die Anschlußeinrichtung (214) anzuhalten, sowie Flußstartsignale, die dem Netzelement erlauben, das Schicken von Datenpaketen an die Anschlußeinrichtung (214) wiederaufzunehmen, enthalten; wobei im Sendeschritt Flußstartsignale nur dann geschickt werden, wenn die Puffereinrichtung (310, 340) ausreichenden Raum für den Empfang eines vollständigen Rundsende-Datenpakets besitzt.

16. Verfahren nach Anspruch 15, ferner dadurch gekennzeichnet, daß

bei jeder der Anschlußeinrichtungen (214) Flußsteuersignale empfangen werden, die von einem mit der jeweiligen Anschlußeinrichtung (214) gekoppelten Netzelement geschickt werden, und die Übertragung des Datenpakets angehalten wird, wenn ein Flußanhaltesignal vom Netzelement empfangen wird, sofern das Datenpaket kein Rundsendepaket ist.

Revendications

1. Réseau local maillé pour l'interconnexion d'une multiplicité d'hôtes, ledit réseau comportant:

une multiplicité de moyens formant commutateur (124, 126, 140, 142, 210) pour acheminer simultanément une multiplicité de paquets de données entre des hôtes (120, 132, 134, 136, 138) se trouvant dans le réseau, chaque moyen formant commutateur (210) comprenant une multiplicité de moyens formant point d'accès (214) pour coupler les moyens formant commutateur à d'autres moyens formant commutateur et à d'autres hôtes, et un commutateur à barres croisées non bloquant (212) pour coupler et acheminer simultanément des paquets de données entre une multiplicité de paires sélectionnées desdits moyens formant point d'accès (214); lesdits hôtes et lesdits moyens formant commutateur formant ensemble des membres de réseau; et
une multiplicité de moyens formant liaison point à point (128) pour interconnecter lesdits moyens formant commutateur et les hôtes dudit réseau, chaque moyen formant liaison point à point (128) fournissant un canal de communication entre deux desdits membres de réseau;
chacun desdits moyens formant point d'accès (214) comprenant des moyens formant mémoire tampon (310, 340) pour la mise en mémoire tampon de paquets de données reçus par lesdits moyens formant commutateur au niveau desdits moyens formant point d'accès (214) et des moyens d'acheminement immédiat (392) pour commencer à retransmettre un paquet de données reçu à travers ledit commutateur à barres croisées non bloquant avant la réception de la fin dudit paquet de données reçu;

caractérisé en ce que:

ladite multiplicité de moyens formant liaison (128) comprend des liaisons d'arbre recouvrant et une multiplicité de liaisons additionnelles;
ladite multiplicité de moyens formant commutateur (124, 126, 140, 142) et lesdites liaisons d'arbre recouvrant constituent ensemble un arbre recouvrant dans lequel l'un desdits moyens formant commutateur est désigné comme racine (710) dudit arbre recouvrant;
chaque dit paquet de données a un hôte spécifié auquel ledit paquet de données est envoyé;
chaque dit moyen formant commutateur (124, 126, 140, 142) comprend des moyens d'acheminement (218) pour définir des itinéraires de transmission de paquets de données autorisés à travers ledit réseau; lesdits moyens d'acheminement désignant un premier sous-ensemble desdits moyens formant liaison comme liaisons montantes, et désignant tous les autres dits moyens formant liaison comme liaisons descendantes;
lesdits moyens d'acheminement définissant pour chaque paquet de données reçu un sous-ensemble autorisé des moyens formant liaison (128) couplés auxdits moyens formant commutateur à travers lesquels ledit paquet de données reçu est autorisé à être retransmis, ledit sous-ensemble autorisé prévu pour des paquets de données reçus sur l'une quelconque desdites liaisons montantes étant sélectionné parmi lesdites liaisons montantes et liaisons descendantes couplées auxdits moyens formant commutateur et ledit sous-ensemble autorisé prévu pour des paquets de données reçus sur l'une quelconque desdites liaisons descendantes étant sélectionnée uniquement parmi lesdites liaisons descendantes couplées auxdits moyens formant commutateur, si bien que le chemin par lequel chaque dit paquet de données est transmis à travers ledit réseau se compose de zéro ou de plus de zéro liaison montante suivie de zéro ou de plus de zéro liaison descendante;
lesdits moyens d'acheminement assurant l'acheminement de paquets de données sans blocage à travers ledit réseau local maillé.

2. Réseau local maillé selon la revendication 1, dans lequel:

lesdits moyens formant liaison (128) sont disposées par paires, chaque paire de moyens formant liaison comprenant un moyen formant liaison désigné comme liaison montante pour transmettre des paquets de données d'un premier membre respectif parmi lesdits membres de réseau à un second membre respectif parmi lesdits membres de réseau, et un moyen formant liaison désigné comme liaison descendante pour transmettre des paquets de données dudit second membre respectif parmi lesdits membres de réseau audit premier membre respectif parmi lesdits membres de réseau;
chaque moyen formant point d'accès (214) de chaque dit moyen formant commutateur est couplé à deux moyens formant liaison, dont une liaison montante et une liaison descendante, qui couplent ledit moyen formant commutateur à un autre membre de réseau; et
ledit arbre recouvrant est défini de telle sorte que ledit second membre respectif parmi ledits membres de réseau est défini comme étant plus proche de ladite racine (710) dudit arbre recouvrant que ne l'est ledit premier membre respectif parmi lesdits membres de réseau.

3. Réseau local maillé selon la revendication 1 ou 2, caractérisé, en outre, en ce que:

lesdits moyens formant commutateur comprennent des moyens d'indication d'état de liaison (382, 406) pour générer des signaux de disponibilité de liaison désignant celui des moyens formant liaison (128) qui est disponible pour retransmettre un paquet de données;
lesdits moyens d'acheminement comprennent des moyens de sélection d'itinéraire (430) pour comparer ledit sous-ensemble autorisé desdits moyens formant liaison (128) par l'intermédiaire desquels un paquet de données reçu peut être retransmis avec lesdits signaux de disponibilité de liaison, et pour acheminer ledit paquet de données reçu par l'intermédiaire d'un moyen formant liaison qui est inclus dans ledit sous-ensemble autorisé desdits moyens formant liaison et qui est disponible pour retransmettre un paquet de données.

4. Réseau local maillé selon la revendication 3, lesdits moyens de sélection d'itinéraire (430) comprenant des moyens (435) pour comparer périodiquement ledit sous-ensemble autorisé desdits moyens formant liaison (128) par l'intermédiaire desquels un paquet de données reçu peut être retransmis avec lesdits signaux de disponibilité de liaison jusqu'à ce que l'un des moyens formant liaison désignés disponibles corresponde à un moyen formant liaison inclus dans ledit sous-ensemble autorisé desdits moyens formant liaison, ledit paquet de données reçu étant alors acheminé par l'intermédiaire du moyen formant liaison correspondant.

5. Réseau local maillé selon la revendication 1, 2, 3 ou 4, caractérisé, en outre, en ce que:

lesdits paquets de données comprennent des paquets de données qui sont envoyés à un seul hôte spécifié parmi lesdits hôtes dudit réseau, et des paquets de données à diffusion générale qui sont à envoyer à tous lesdits hôtes dudit réseau;
lesdits moyens d'acheminement comprennent des moyens pour désigner deux itinéraires de transmission de paquets à diffusion générale, comprenant des moyens pour désigner comme point d'accès de liaison montante un moyen formant point d'accès (214) d'un dit moyen formant commutateur qui couple ledit moyen formant commutateur à un autre moyen formant commutateur qui est plus proche de ladite racine dudit arbre recouvrant, conformément à des critères prédéfinis, et pour désigner comme points d'accès de liaison descendante chaque moyen formant point d'accès (214) qui couple, avec des liaisons d'arbre recouvrant, ledit moyen formant commutateur à un autre moyen formant commutateur qui est plus éloigné de ladite racine dudit arbre recouvrant, conformément auxdits critères prédéfinis;
lesdits moyens d'acheminement de chaque dit moyen formant commutateur, à l'exception dudit moyen formant commutateur désigné comme étant ladite racine (710) dudit arbre recouvrant, acheminant des paquets de données à diffusion générale reçus par ledit point d'accès de liaison montante vers tous lesdits points d'accès de liaison descendante, et acheminant vers ledit point d'accès de liaison montante des paquets de données à diffusion générale reçus par l'un quelconque desdits moyens formant point d'accès (214) autre que ledit point d'accès de liaison montante.

6. Réseau local maillé selon la revendication 6, caractérisé, en outre, en ce que:
lesdits moyens d'acheminement dudit moyen formant commutateur désigné comme étant ladite racine dudit arbre recouvrant comprend des moyens pour acheminer tous les paquets de données à diffusion générale reçus par ledit moyen formant commutateur vers tous lesdits points d'accès de liaison descendante.

7. Réseau local maillé selon la revendication 5 ou 6,

lesdits paquets de données à diffusion générale ayant une taille maximale prédéfinie;
lesdits moyens formant mémoire tampon (310, 340) ayant plus qu'assez d'espace pour emmagasiner tout un paquet de données à diffusion générale de ladite taille maximale prédéterminée et des moyens pour l'indiquer quand lesdits moyens formant mémoire tampon (310, 340) ont assez d'espace pour recevoir tout un paquet de données à diffusion générale;
lesdits moyens formant point d'accès (214) comprenant des moyens de commande de flux (406) couplés auxdits moyens formant mémoire tampon (310, 340) pour envoyer des signaux de commande de flux à un membre de réseau couplé auxdits moyens formant point d'accès (214), lesdits signaux de commande de flux comprenant des signaux d'arrêt de flux exigeant que ledit membre de réseau cesse de transmettre des paquets de données auxdits moyens formant point d'accès (214) et des signaux de déclenchement de flux autorisant ledit membre de réseau à recommencer à envoyer des paquets de données auxdits moyens formant point d'accès (214); lesdits moyens de commande de flux envoyant des signaux de déclenchement de flux uniquement lorsque lesdits moyens formant mémoire tampon (310, 340) ont assez d'espace pour recevoir tout un paquet de données à diffusion générale.

8. Réseau local maillé selon la revendication 7, lesdits moyens formant point d'accès (214) comprenant des moyens (392) pour recevoir lesdits signaux de commande de flux envoyés par un membre de réseau couplé auxdits moyens formant point d'accès (214) et pour mettre fin à la transmission d'un paquet de données lorsqu'un signal d'arrêt de flux est reçu dudit membre de réseau, à moins que ledit paquet de données ne soit un paquet à diffusion générale.

9. Procédé d'exploitation d'un réseau local maillé, le réseau comprenant:

une multiplicité de moyens formant commutateur (124, 126, 140, 142, 210) pour acheminer simultanément une multiplicité de paquets de données entre des hôtes (120, 132, 134, 136, 138) se trouvant dans le réseau, chaque moyen formant commutateur (210) comprenant une multiplicité de moyens formant point d'accès (214) pour coupler les moyens formant commutateur à d'autres moyens formant commutateur et et à d'autres hôtes, et un commutateur à barres croisées non bloquant (212) pour coupler et acheminer simultanément des paquets de données entre une multiplicité de paires sélectionnées desdits moyens formant point d'accès (214); lesdits hôtes et lesdits moyens formant commutateur formant ensemble des membres de réseau; et
une multiplicité de moyens formant liaison point à point (128) pour interconnecter lesdits moyens formant commutateur et les hôtes dudit réseau, chaque moyen formant liaison point à point (128) fournissant un canal de communication entre deux desdits membres de réseau;
chacun desdits moyens formant point d'accès (214) comprenant des moyens formant mémoire tampon (310, 340) pour la mise en mémoire tampon de paquets de données reçus par lesdits moyens formant commutateur au niveau desdits moyens formant point d'accès (214) et des moyens d'acheminement immédiat (392) pour commencer à retransmettre un paquet de données reçu à travers ledit commutateur à barres croisées non bloquant avant la réception de la fin dudit paquet de données reçu;

ledit procédé comportant les étapes consistant:
au niveau de chaque dit moyen formant commutateur, lors de la réception d'une partie initiale d'un paquet de données, lorsque des critères de disponibilité de liaison prédéfinis sont remplis, à commencer la retransmission dudit paquet de données reçu à travers ledit commutateur à barres croisées non bloquant avant que la fin dudit paquet de données reçu n'ait été reçue, et, sinon, à mettre en mémoire tampon ledit paquet de données reçu jusqu'à ce que lesdits critères de disponibilité de liaison prédéfinis soient remplis;
caractérisé par:

la configuration desdits membres de réseau comme arbre recouvrant, l'un desdits moyens formant commutateur étant désigné comme racine (710) dudit arbre recouvrant, et ladite multiplicité de moyens formant liaison (128) comprenant des liaisons d'arbre recouvrant qui font partie dudit arbre recouvrant, et une multiplicité de liaisons additionnelles, qui ne font pas partie dudit arbre recouvrant;
la définition d'itinéraires de transmission de paquets de données autorisés à travers ledit réseau, y compris la désignation d'un premier sous-ensemble desdits moyens formant liaison comme liaisons montantes et la désignation de tous les autres dits moyens formant liaison comme liaisons descendantes;
au niveau de chacun desdits moyens formant commutateur, la définition, pour chaque paquet de données reçu, d'un sous-ensemble autorisé des moyens formant liaison (128) couplés auxdits moyens formant commutateur par l'intermédiaire desquels ledit paquet de données reçu est autorisé à être retransmis, ledit sous-ensemble autorisé prévu pour des paquets de données reçus sur l'une quelconque desdites liaisons montantes

étant sélectionné parmi lesdites liaisons montantes et liaisons descendantes couplées auxdits moyens formant commutateur et ledit sous-ensemble autorisé prévu pour des paquets de données reçus sur l'une quelconque desdites liaisons descendantes étant sélectionnée uniquement parmi lesdites liaisons descendantes couplées auxdits moyens formant commutateur, et l'acheminement de chaque paquet de données reçu par l'intermédiaire d'un moyen formant liaison inclus dans ledit sous-ensemble autorisé défini de moyens formant liaison, si bien que le chemin par lequel chaque dit paquet de données est transmis à travers ledit réseau se compose de zéro ou de plus de zéro liaison montante suivie de zéro ou de plus de zéro liaison descendante;
lesdites étapes de définition et d'acheminement assurant l'acheminement de paquets de données sans blocage à travers ledit réseau local maillé.

10. Procédé selon la revendication 9, comprenant, en outre, les étapes consistant:

à disposer lesdits moyens formant liaison (128) par paires, chaque paire de moyens formant liaison comprenant un moyen formant liaison désigné comme liaison montante pour transmettre des paquets de données d'un premier membre respectif parmi lesdits membres de réseau à un second membre respectif parmi lesdits membres de réseau, et un moyen formant liaison désigné comme liaison descendante pour transmettre des paquets de données dudit second membre respectif parmi lesdits membres de réseau audit premier membre respectif parmi lesdits membres de réseau;
à coupler chaque moyen formant point d'accès (214) de chaque dit moyen formant commutateur à deux moyens formant liaison, dont une liaison montante et une liaison descendante, qui couplent ledit moyen formant commutateur à un autre membre de réseau; et
à définir ledit arbre recouvrant de telle sorte que ledit second membre respectif parmi ledits membres de réseau soit défini comme étant plus proche de ladite racine (710) dudit arbre recouvrant que ne l'est ledit premier membre respectif parmi lesdits membres de réseau.

11. Procédé selon la revendication 9 ou 10, caractérisé, en outre, par:

au niveau de chaque moyen formant commutateur (382, 406), la génération de signaux de disponibilité de liaison désignant celui des moyens formant liaison (128) qui est disponible pour retransmettre un paquet de données;
le fait que ladite étape d'acheminement consiste, entre autres, à comparer ledit sous-ensemble autorisé desdits moyens formant liaison, par l'intermédiaire desquels un paquet de données reçu peut être retransmis, auxdits signaux de disponibilité de liaison, et à acheminer ledit paquet de données reçu par l'intermédiaire d'un moyen formant liaison qui est inclus dans ledit sous-ensemble autorisé desdits moyens formant liaison et qui est disponible pour retransmettre un paquet de données.

12. Procédé selon la revendication 11, caractérisé, en outre en ce que:
ladite étape d'acheminement consiste, entre autres, à comparer périodiquement ledit sous-ensemble autorisé desdits moyens formant liaison (128), par l'intermédiaire desquels un paquet de données reçu peut être retransmis, auxdits signaux de disponibilité de liaison jusqu'à ce que l'un des moyens formant liaison désignés disponibles corresponde à un moyen formant liaison inclus dans ledit sous-ensemble autorisé desdits moyens formant liaison, puis à acheminer ledit paquet de données reçu par l'intermédiaire du moyen formant liaison correspondant.

13. Procédé selon la revendication 9, 10, 11 ou 12, caractérisé, en outre, en ce que:

lesdits paquets de données comprennent des paquets de données qui sont envoyés à un seul hôte spécifié parmi lesdits hôtes dudit réseau, et des paquets de données à diffusion générale qui sont à envoyer à tous lesdits hôtes dudit réseau;
ladite étape de définition d'itinéraires de transmission de paquets de données autorisés comprend la désignation de deux itinéraires de transmission de paquets à diffusion générale, y compris la désignation comme point d'accès de liaison montante un moyen formant point d'accès (214) d'un dit moyen formant commutateur qui couple ledit moyen formant commutateur à un autre moyen formant commutateur qui est plus proche de ladite racine (710) dudit arbre recouvrant, conformément à des critères prédéfinis, et la désignation comme points d'accès de liaison descendante chaque moyen formant point d'accès (214) qui couple, avec des liaisons d'arbre recouvrant, ledit moyen formant commutateur à un autre moyen formant commutateur qui est plus éloigné de ladite racine (710) dudit arbre recouvrant, conformément auxdits critères prédéfinis;
ladite étape d'acheminement, telle qu'exécutée par chaque dit moyen formant commutateur, à l'exception

dudit moyen formant commutateur désigné comme étant ladite racine (710) dudit arbre recouvrant, consiste, entre autres, à acheminer des paquets de données à diffusion générale reçus par ledit point d'accès de liaison montante vers tous lesdits points d'accès de liaison descendante, et à acheminer vers ledit point d'accès de liaison montante des paquets de données à diffusion générale reçus par l'un quelconques desdits moyens formant point d'accès (214) autre que ledit point d'accès de liaison montante.

14. Procédé selon la revendication 13, caractérisé, en outre, en ce que:

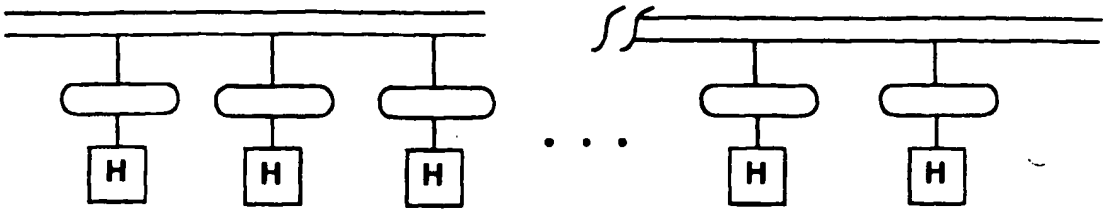
ladite étape d'acheminement, telle qu'exécutée par ledit moyen formant commutateur désigné comme étant ladite racine (710) dudit arbre recouvrant, consiste, entre autres, à acheminer tous les paquets de données à diffusion générale reçus par ledit moyen formant commutateur à tous lesdits points d'accès de liaison descendante.

15. Procédé selon la revendication 12 ou 13, caractérisé, en outre, par:

le fait que lesdits paquets de données à diffusion générale ont une taille maximale prédéfinie;
la dotation de chacun desdits moyens formant point d'accès (214) d'un moyen formant mémoire tampon (310, 340) ayant plus qu'assez d'espace pour emmagasiner tout un paquet de données à diffusion générale de ladite taille maximale prédéterminée et des moyens pour l'indiquer quand lesdits moyens formant mémoire tampon (310, 340) ont assez d'espace pour recevoir tout un paquet de données à diffusion générale;
au niveau de chaque dit moyen formant point d'accès (214), l'envoi de signaux de commande de flux à un membre de réseau couplé auxdits moyens formant point d'accès (214), lesdits signaux de commande de flux comprenant des signaux d'arrêt de flux exigeant que ledit membre de réseau cesse de transmettre des paquets de données auxdits moyens formant point d'accès (214) et des signaux de déclenchement de flux autorisant ledit membre de réseau à recommencer à envoyer des paquets de données auxdits moyens formant point d'accès (214); ladite étape d'envoi consistant à envoyer des signaux de déclenchement de flux uniquement lorsque lesdits moyens formant mémoire tampon (310, 340) ont assez d'espace pour recevoir tout un paquet de données à diffusion générale.

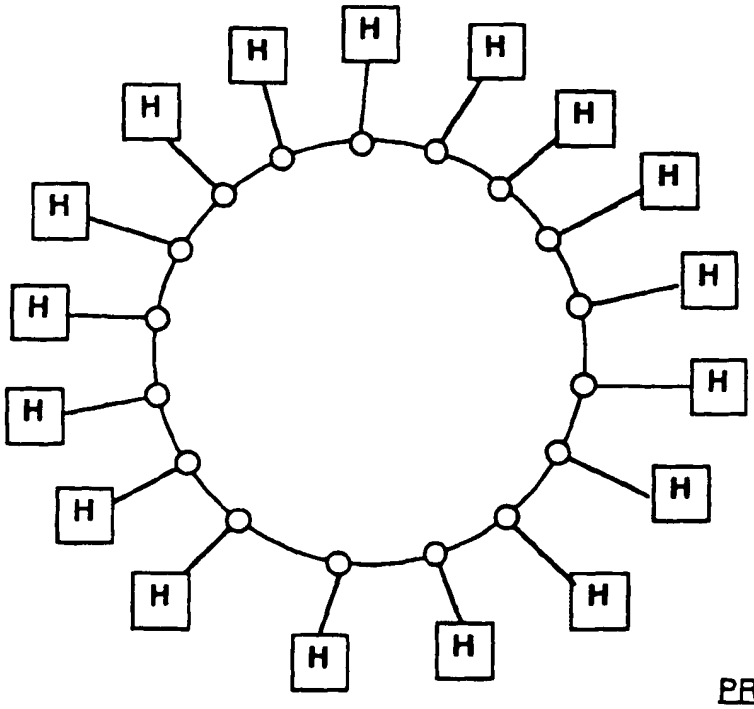
16. Procédé selon la revendication 15, caractérisé, en outre, par:

au niveau de chaque dit moyen formant point d'accès (214), la réception desdits signaux de commande de flux envoyés par un membre de réseau couplé auxdits moyens formant point d'accès (214), et l'arrêt de la transmission d'un paquet de données lorsqu'un signal d'arrêt de flux est reçu dudit membre de réseau, à moins que ledit paquet de données ne soit un paquet à diffusion générale.



PRIOR ART

FIGURE 1A



PRIOR ART

FIGURE 1B

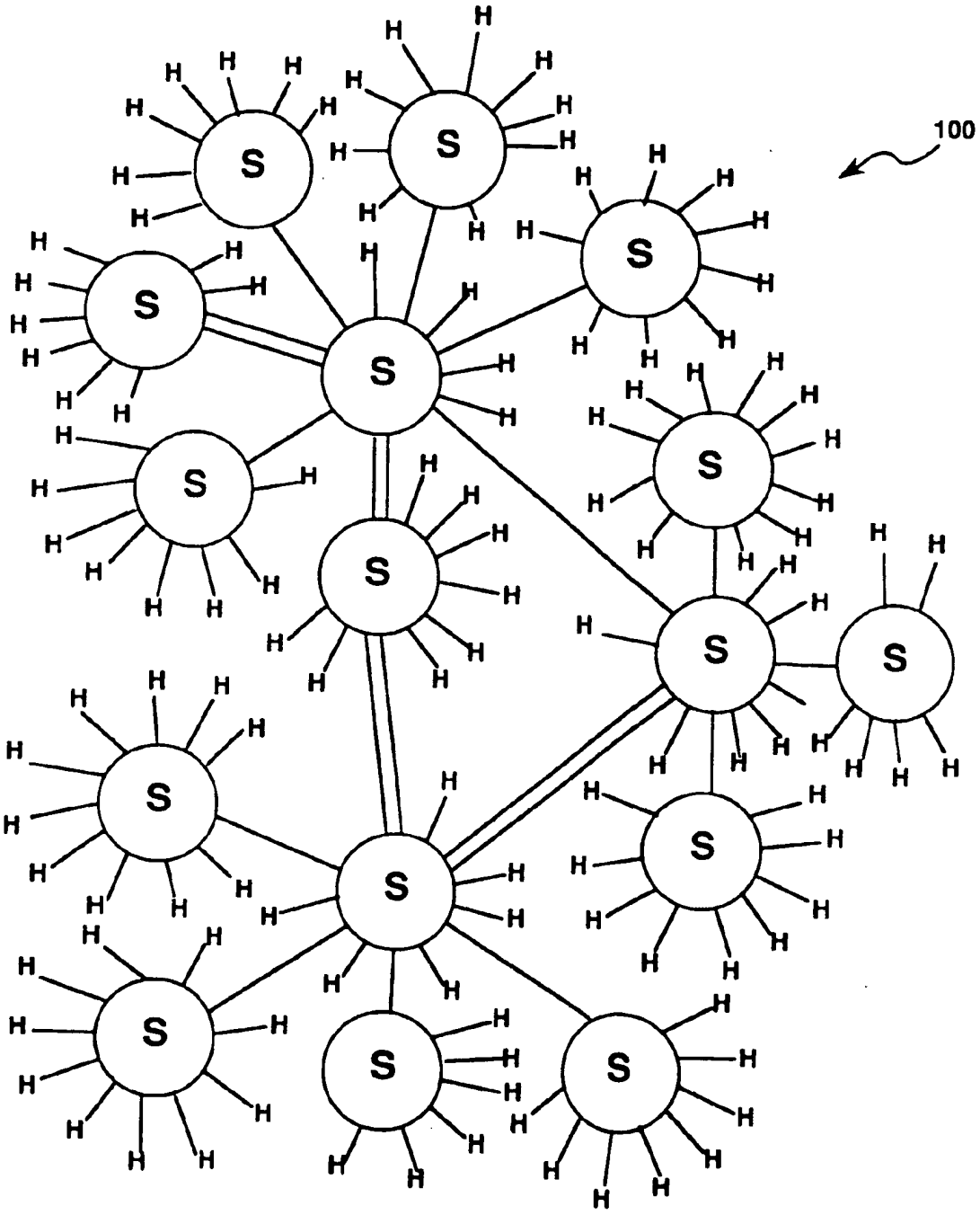


FIGURE 2

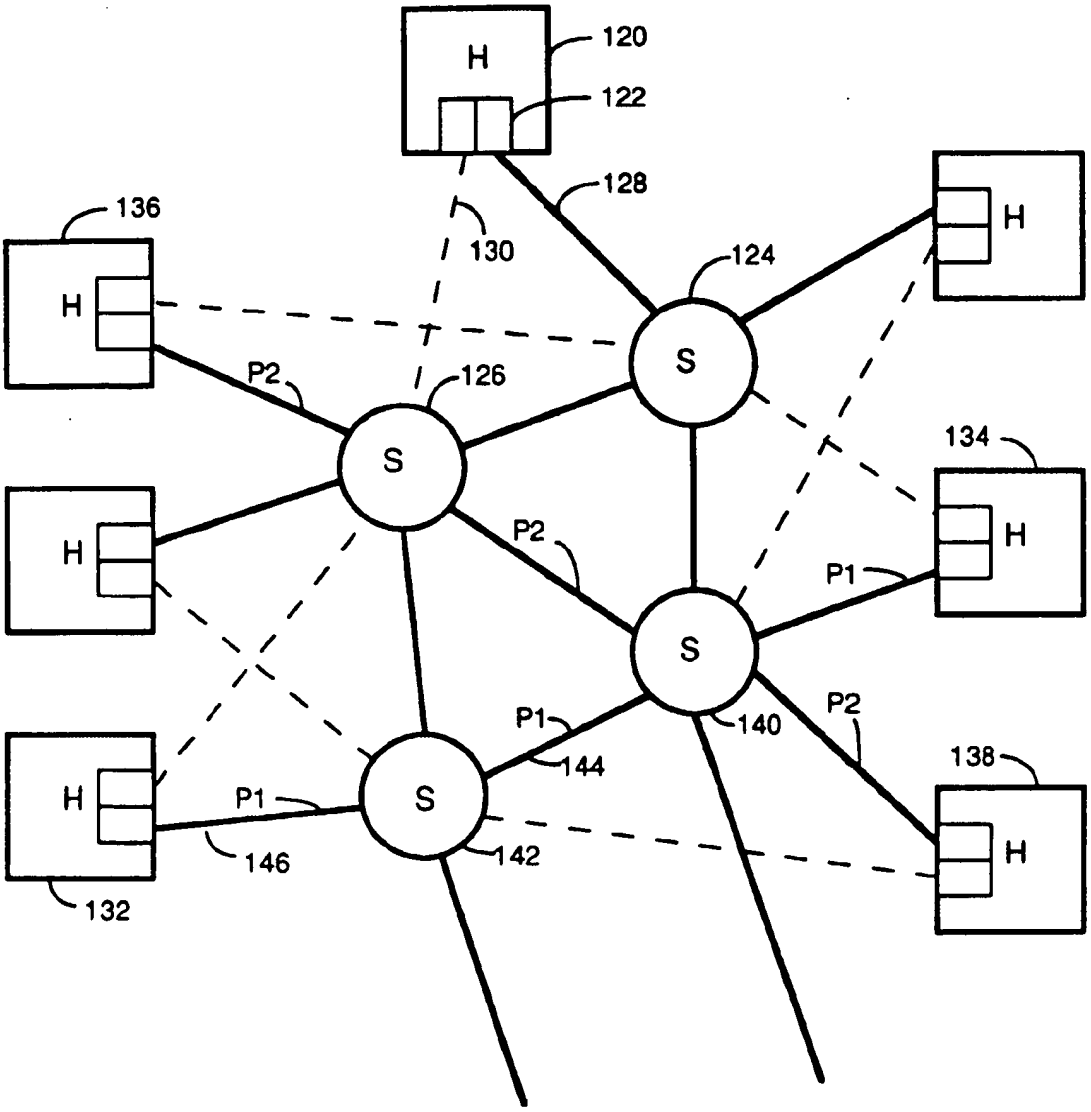

H
120
122
128
130
136
H
124
H
P2
126
S
S
134
H
H
P2
P1
S
P1
140
P2
144
138
H
P1
S
H
146
142
132

FIGURE 3

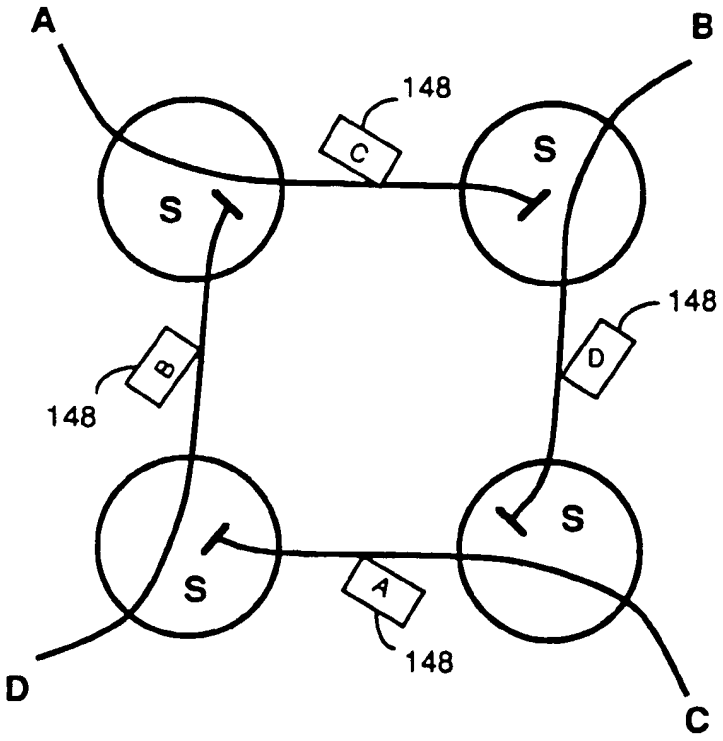


FIGURE 4

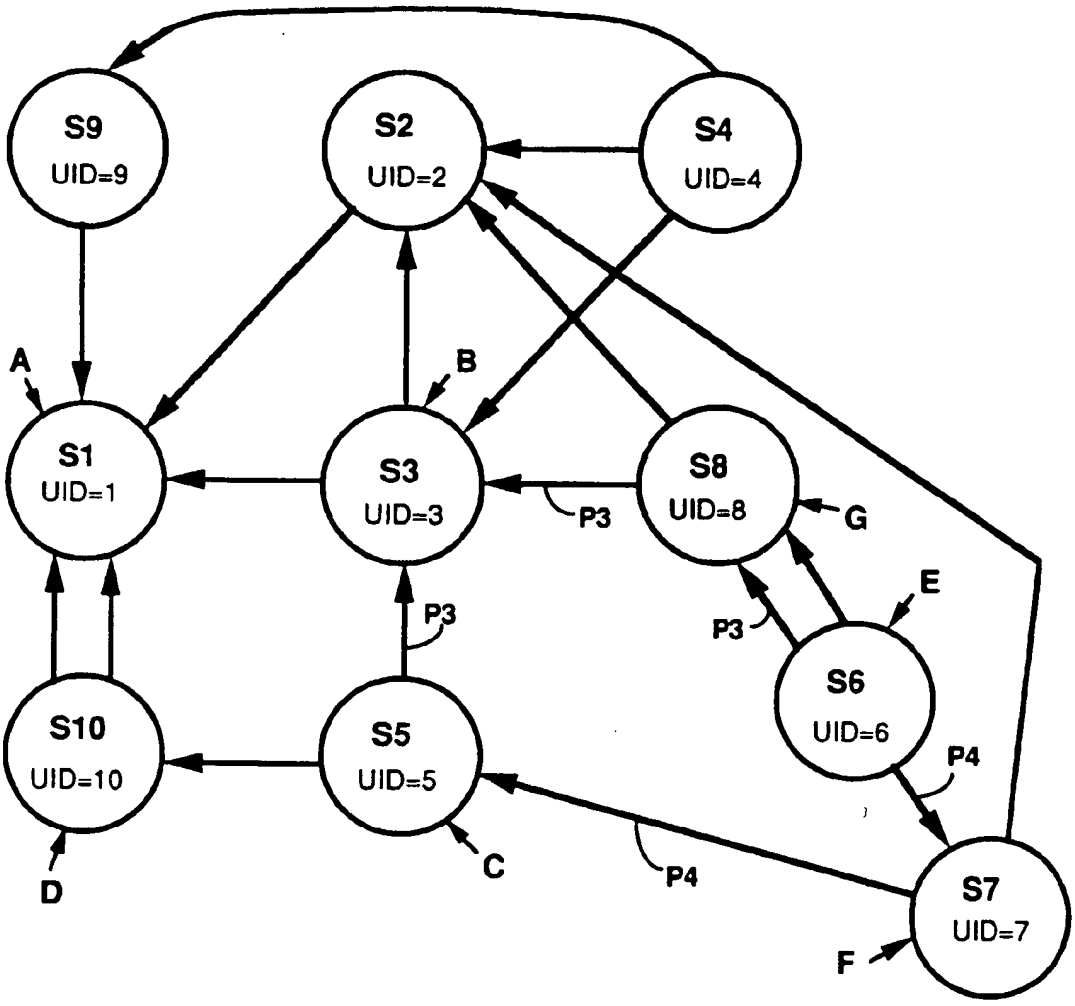

S9
UID=9
S2
UID=2
S4
UID=4
A
B
S1
UID=1
S3
UID=3
S8
UID=8
P3
G
E
P3
P3
S6
UID=6
S10
UID=10
S5
UID=5
P4
C
P4
S7
UID=7
D
F

FIGURE 5

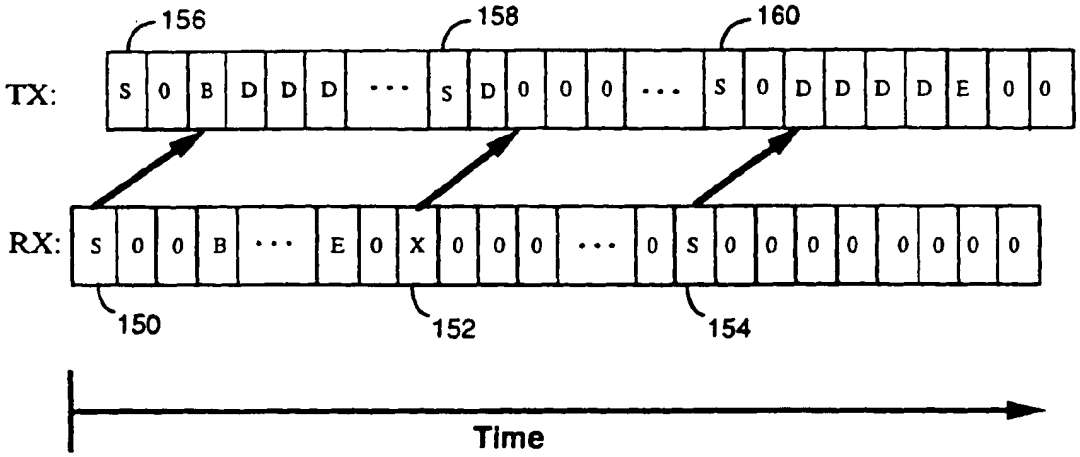

156
158
160
TX:
S 0 B D D D ... S D 0 0 0 ... S 0 D D D D E 0 0
RX:
S 0 0 B ... E 0 X 0 0 0 ... 0 S 0 0 0 0 0 0 0 0
150
152
154
Time

FIGURE 6

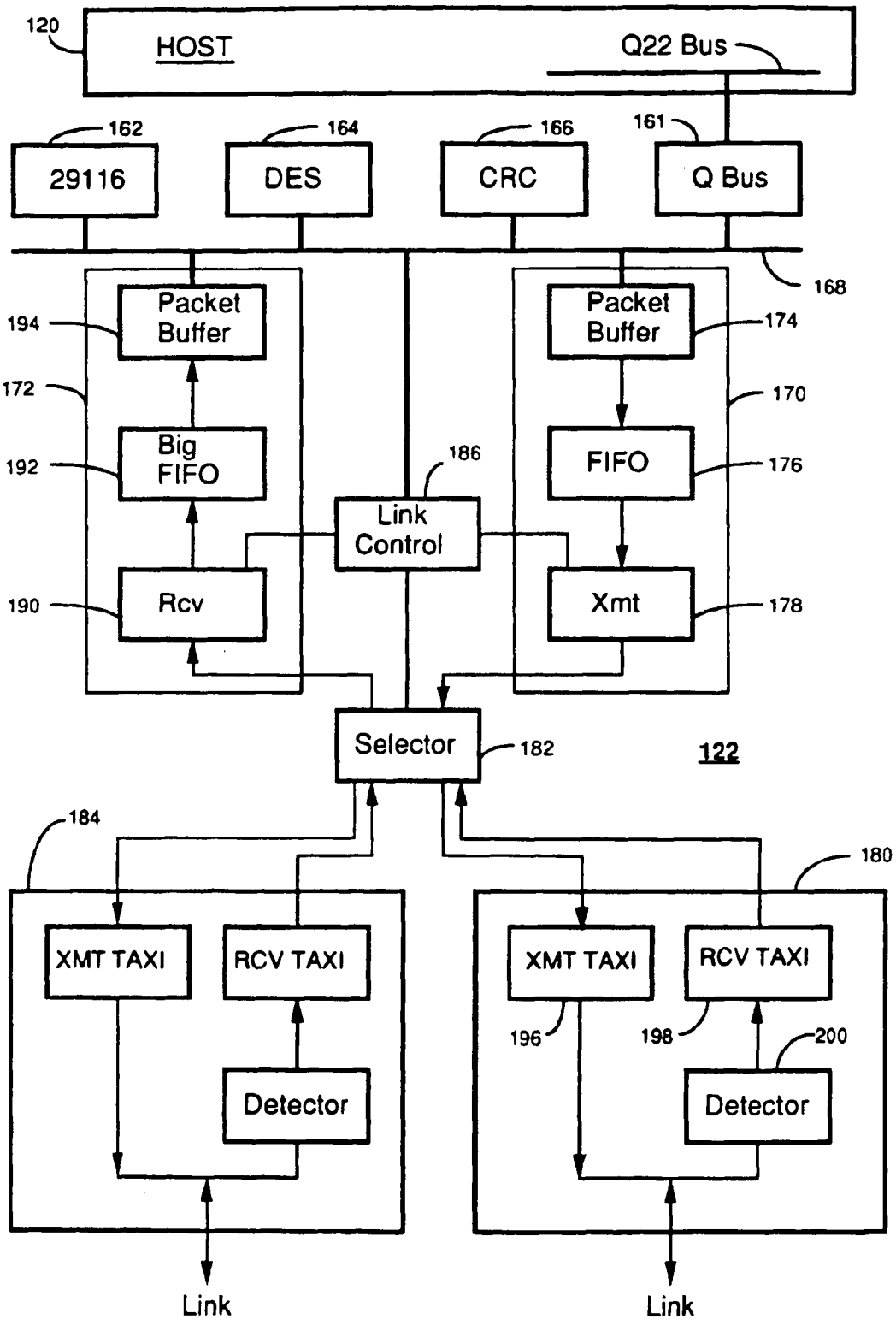

120
HOST
Q22 Bus
162
29116
164
DES
166
CRC
161
Q Bus
168
194
Packet Buffer
172
Big FIFO
192
190
Rcv
186
Link Control
Packet Buffer
174
170
FIFO
176
Xmt
178
Selector
182
122
184
180
XMT TAXI
RCV TAXI
Detector
196
198
200
XMT TAXI
RCV TAXI
Detector
Link
Link

FIGURE 7

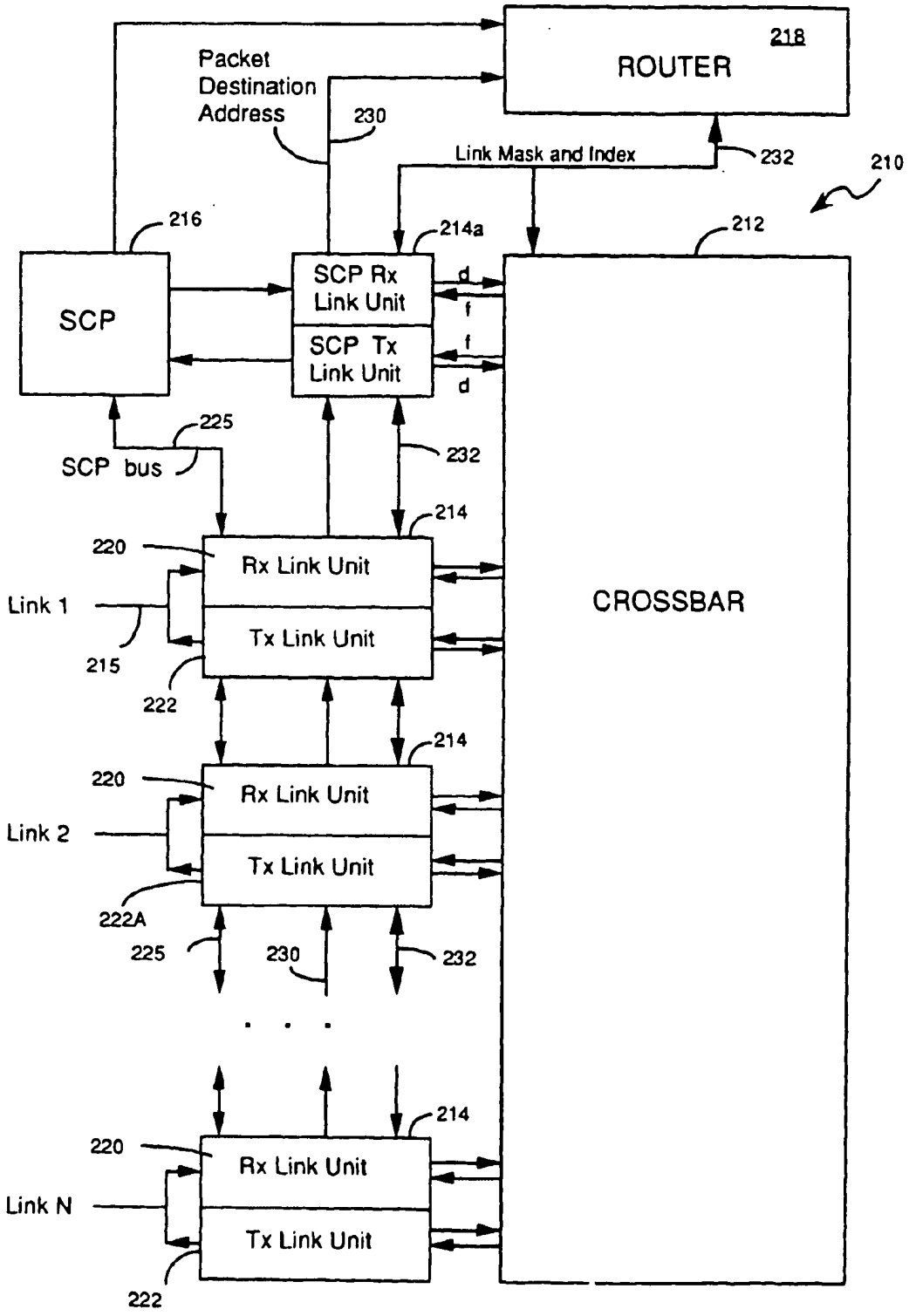

ROUTER
218
Packet
Destination
Address
230
Link Mask and Index
232
210
216
214a
212
SCP
SCP Rx
Link Unit
SCP Tx
Link Unit
d
f
f
d
225
SCP bus
232
214
220
Rx Link Unit
Link 1
Tx Link Unit
215
222
CROSSBAR
214
220
Rx Link Unit
Link 2
Tx Link Unit
222A
225
230
232
214
220
Rx Link Unit
Link N
Tx Link Unit
222

FIGURE 8

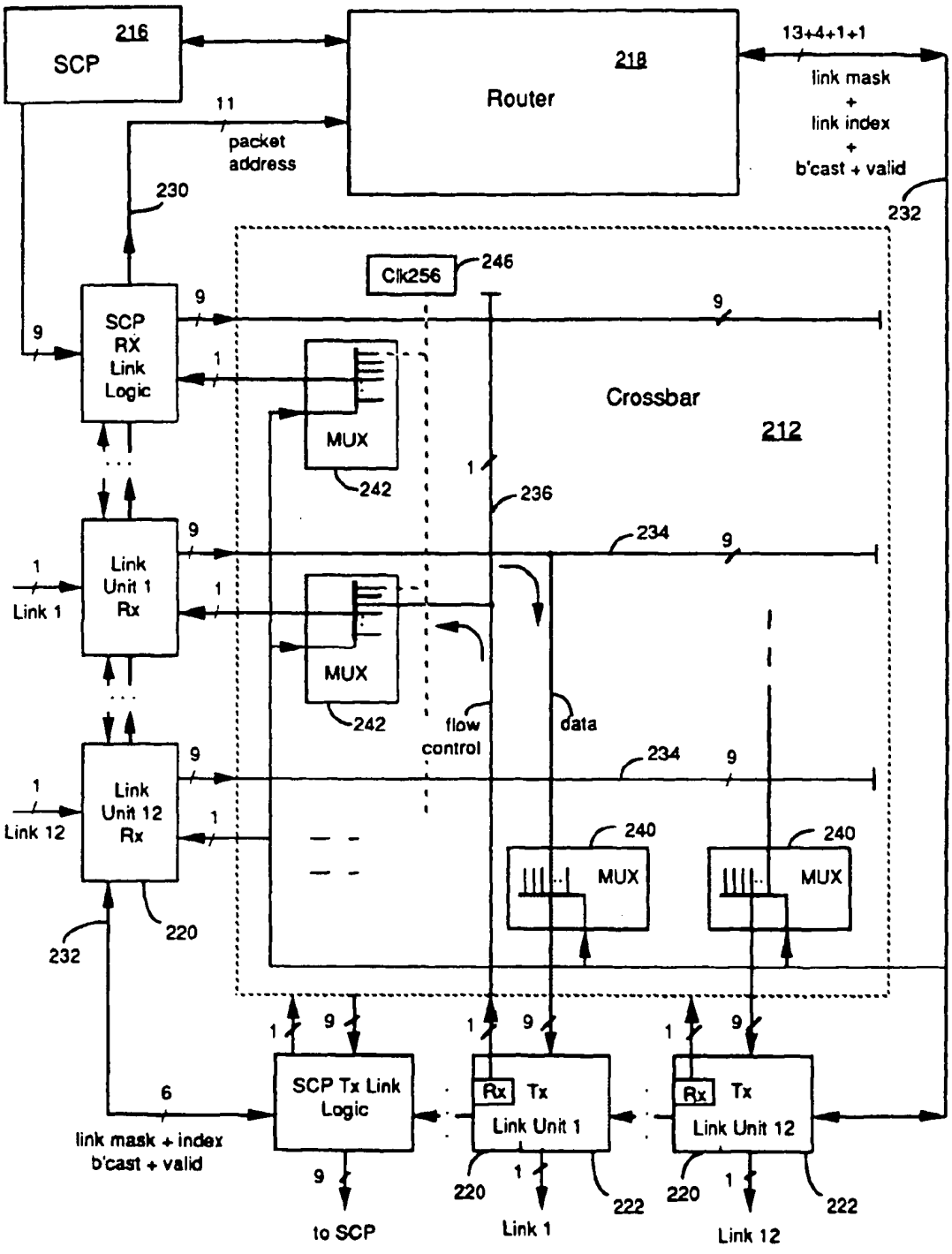

SCP
216
Router
218
13+4+1+1
link mask
+
link index
+
b'cast + valid
232
11
packet
address
230
Clk256
246
SCP
RX
Link
Logic
9
1
MUX
242
Crossbar
212
236
234
Link
Unit 1
Rx
Link 1
Link
Unit 12
Rx
Link 12
flow
control
data
240
220
SCP Tx Link
Logic
6
link mask + index
b'cast + valid
Rx
Tx
Link Unit 1
Link Unit 12
to SCP
Link 1
Link 12
222

FIGURE 9

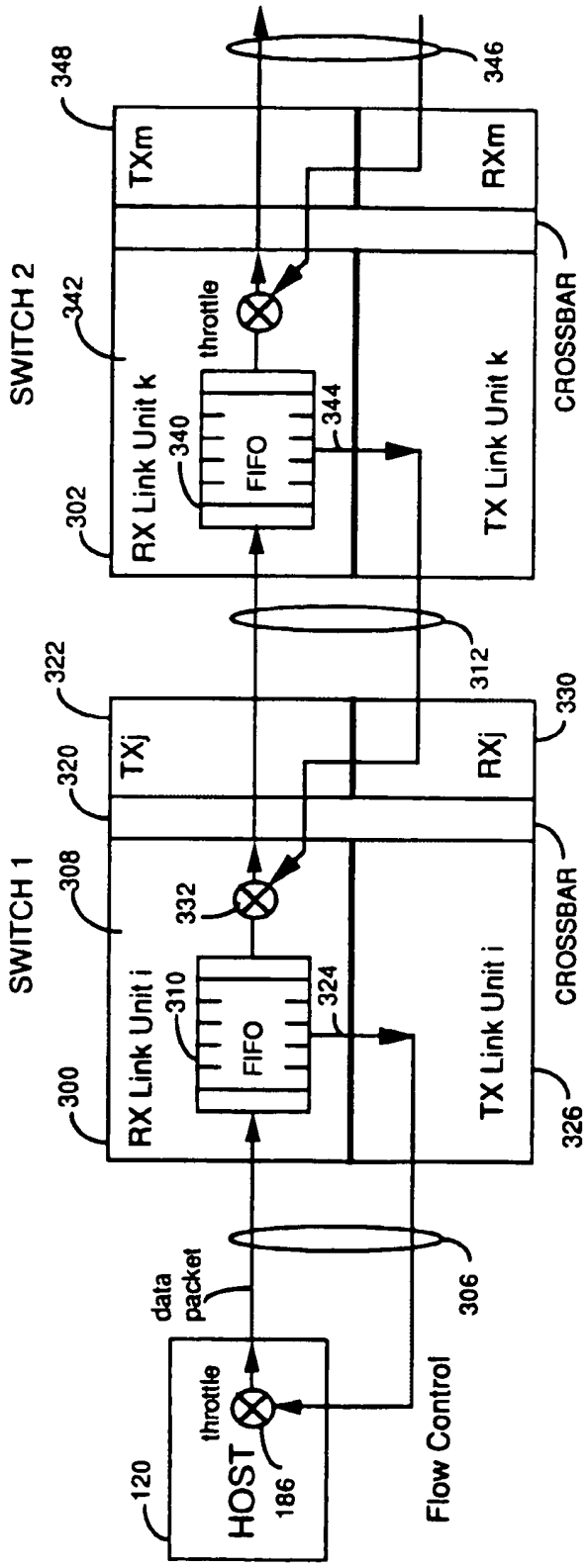

SWITCH 2
SWITCH 1
RX Link Unit k
RX Link Unit i
TX Link Unit k
TX Link Unit i
TXm
RXm
TXj
RXj
FIFO
throttle
CROSSBAR
HOST
data packet
Flow Control
120
186
300
302
306
308
310
312
320
322
324
326
330
332
340
342
344
346
348

FIGURE 10

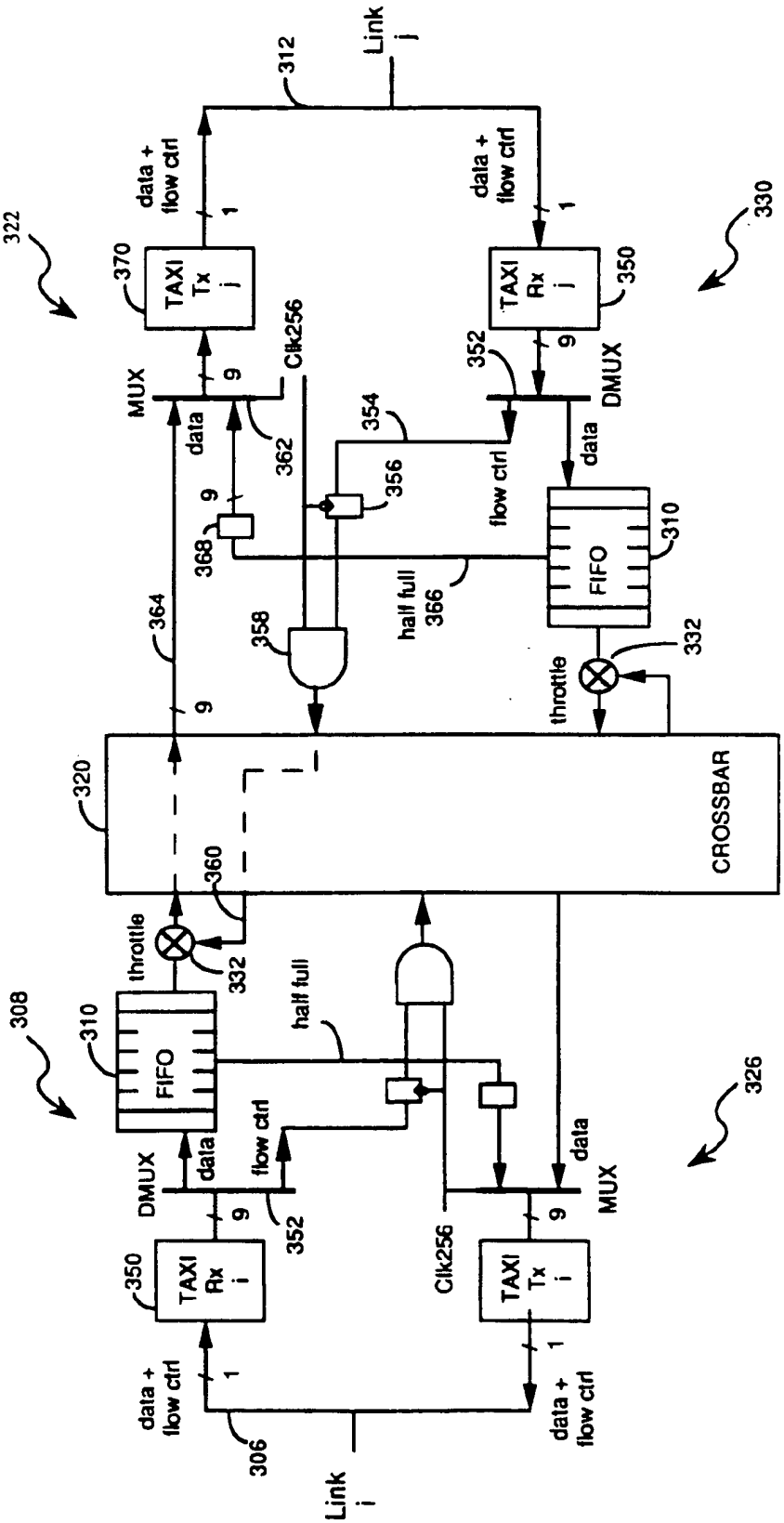

Link j
312
data + flow ctrl
1
322
330
370
TAXI Tx j
TAXI Rx j
350
MUX
9
Clk256
352
DMUX
data
362
354
356
flow ctrl
368
half full
366
FIFO
310
364
358
332
throttle
CROSSBAR
320
360
308
326
DMUX
MUX
TAXI Rx i
TAXI Tx i
306
Link i

FIGURE 11

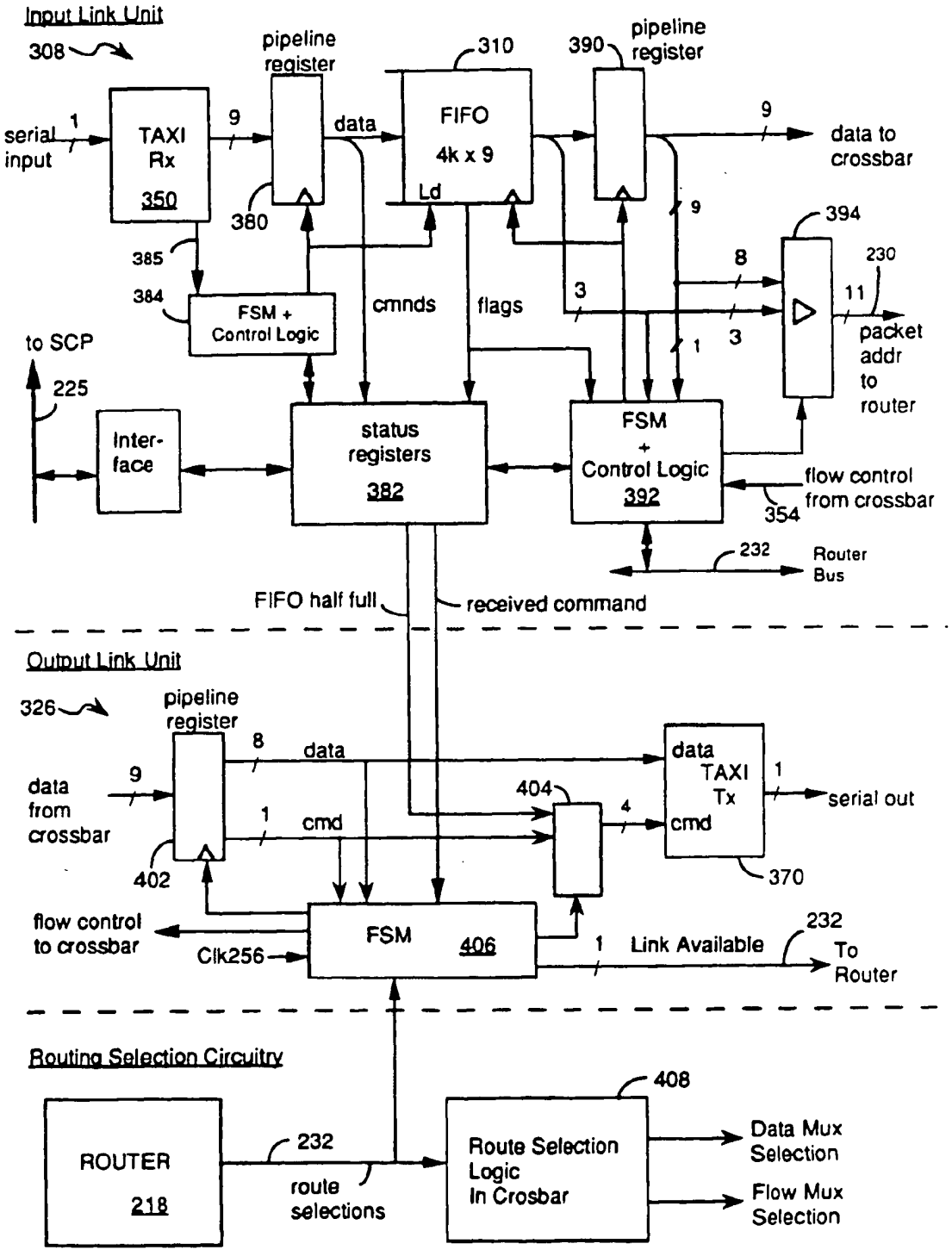

Input Link Unit
308
pipeline register
310
390
pipeline register
serial input
1
TAXI Rx
350
9
data
FIFO
4k x 9
Ld
9
data to crossbar
380
9
394
385
8
230
384
FSM + Control Logic
cmnds
flags
3
11
packet addr to router
1
3
to SCP
225
Inter-face
status registers
382
FSM + Control Logic
392
flow control from crossbar
354
232
Router Bus
FIFO half full
received command
Output Link Unit
326
pipeline register
8
data
data TAXI Tx
1
serial out
data from crossbar
9
404
4
1
cmd
cmd
402
370
flow control to crossbar
FSM
406
232
Clk256
1
Link Available
To Router
Routing Selection Circuitry
408
ROUTER
218
232
route selections
Route Selection Logic In Crosbar
Data Mux Selection
Flow Mux Selection

FIGURE 12

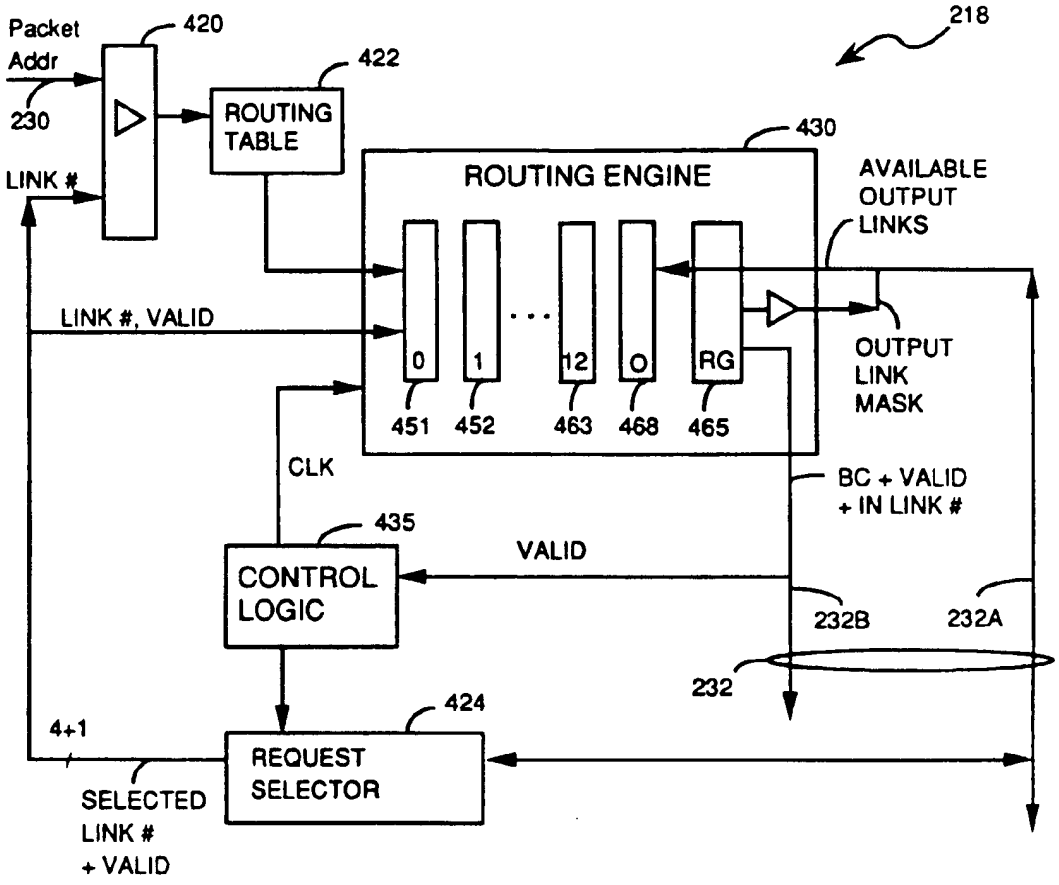

Packet
Addr
230
LINK #
420
422
ROUTING
TABLE
218
430
ROUTING ENGINE
AVAILABLE
OUTPUT
LINKS
LINK #, VALID
0
1
12
O
RG
451
452
463
468
465
OUTPUT
LINK
MASK
CLK
BC + VALID
+ IN LINK #
435
CONTROL
LOGIC
VALID
232B
232A
232
424
4+1
REQUEST
SELECTOR
SELECTED
LINK #
+ VALID

FIGURE 13

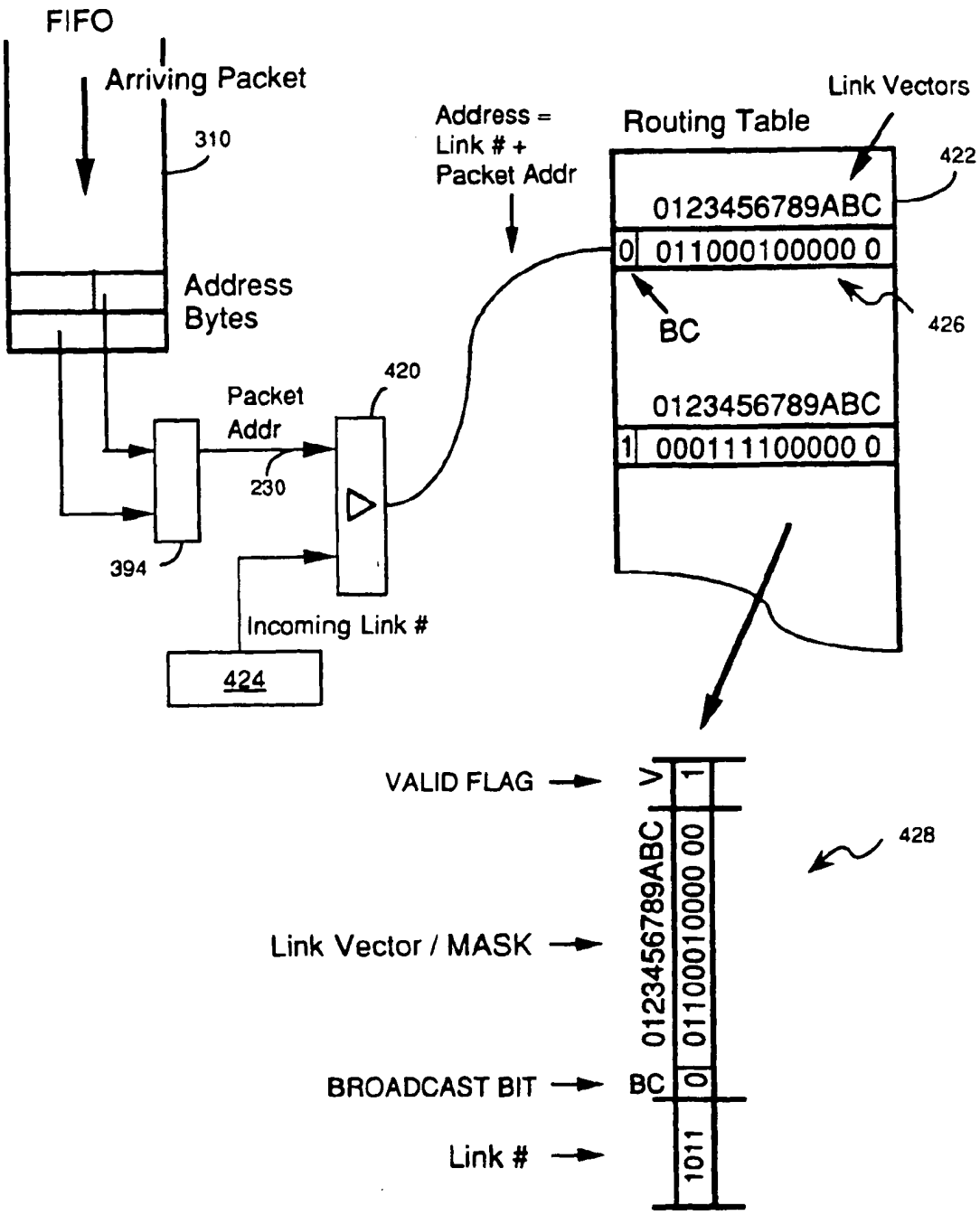

FIFO
Arriving Packet
310
Address
Bytes
Packet
Addr
230
420
394
Incoming Link #
424
Address =
Link # +
Packet Addr
Routing Table
Link Vectors
422
0123456789ABC
0 011000100000 0
BC
426
0123456789ABC
1 000111100000 0
VALID FLAG
V
1
428
Link Vector / MASK
0123456789ABC
011000100000 00
BROADCAST BIT
BC
0
Link #
1011

FIGURE 14

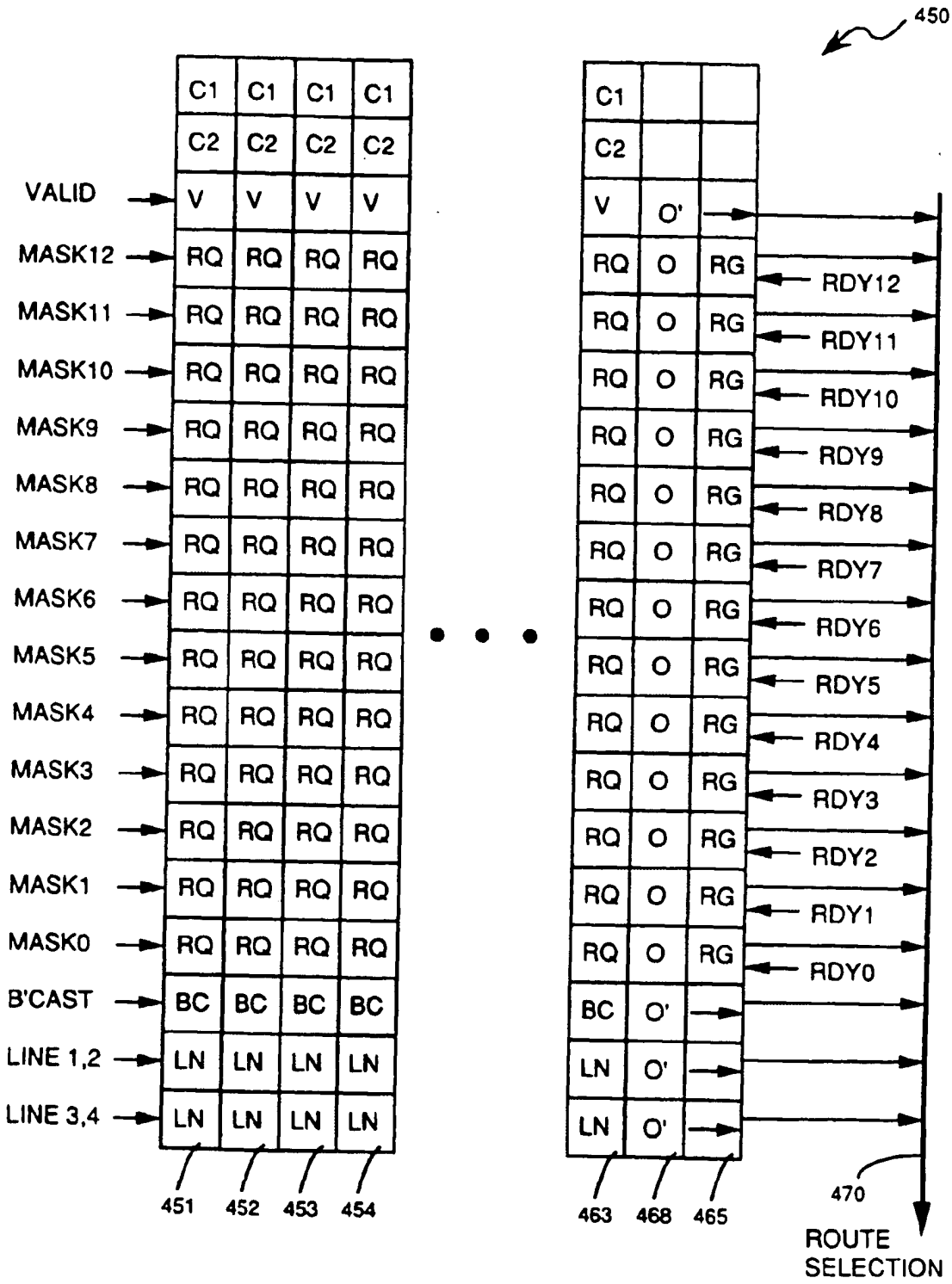

450
C1 C1 C1 C1
C2 C2 C2 C2
VALID V V V V
MASK12 RQ RQ RQ RQ
MASK11 RQ RQ RQ RQ
MASK10 RQ RQ RQ RQ
MASK9 RQ RQ RQ RQ
MASK8 RQ RQ RQ RQ
MASK7 RQ RQ RQ RQ
MASK6 RQ RQ RQ RQ
MASK5 RQ RQ RQ RQ
MASK4 RQ RQ RQ RQ
MASK3 RQ RQ RQ RQ
MASK2 RQ RQ RQ RQ
MASK1 RQ RQ RQ RQ
MASK0 RQ RQ RQ RQ
B'CAST BC BC BC BC
LINE 1,2 LN LN LN LN
LINE 3,4 LN LN LN LN
451 452 453 454
C1
C2
V O'
RQ O RG RDY12
RQ O RG RDY11
RQ O RG RDY10
RQ O RG RDY9
RQ O RG RDY8
RQ O RG RDY7
RQ O RG RDY6
RQ O RG RDY5
RQ O RG RDY4
RQ O RG RDY3
RQ O RG RDY2
RQ O RG RDY1
RQ O RG RDY0
BC O'
LN O'
LN O'
463 468 465
470
ROUTE SELECTION

FIGURE 15

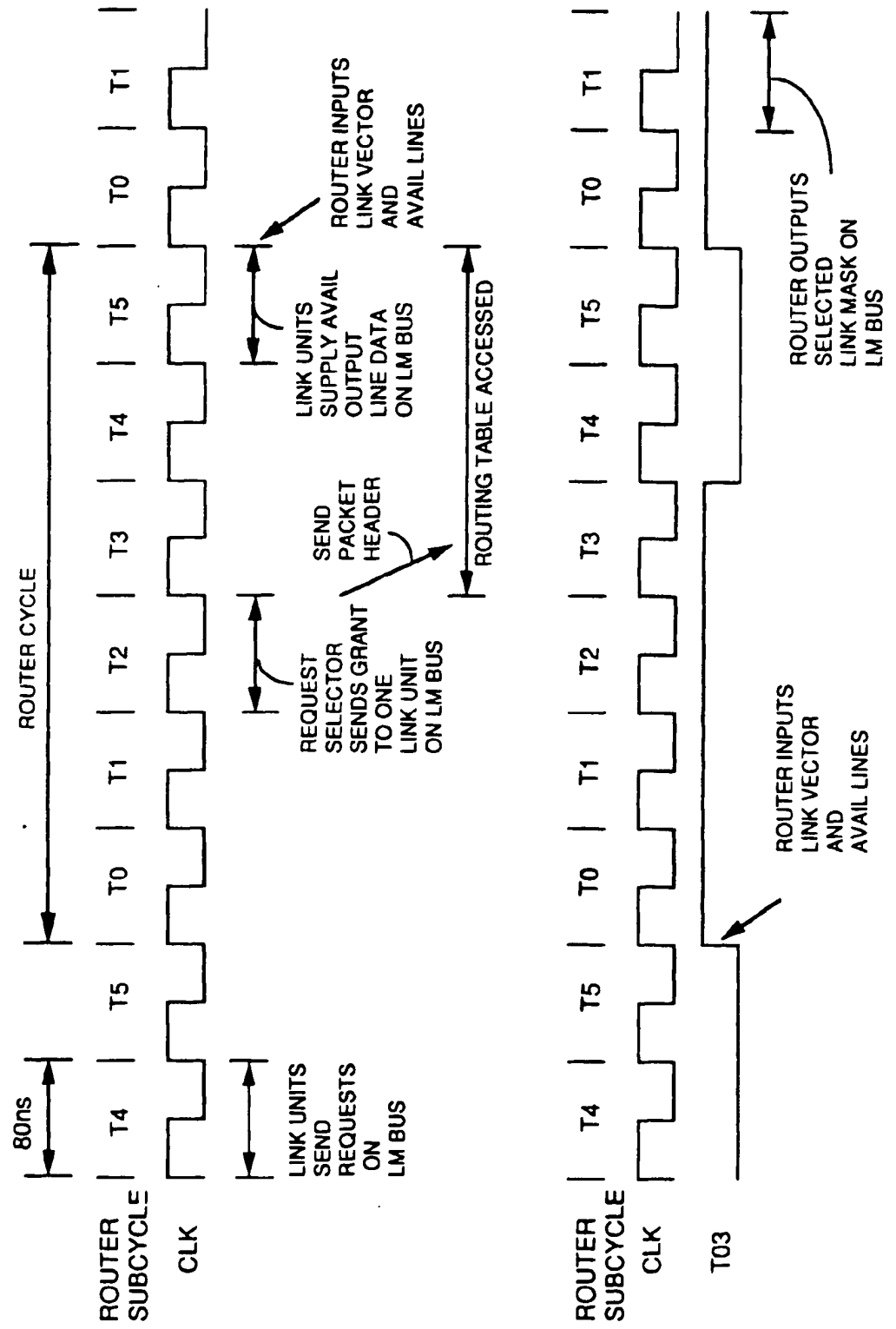

ROUTER SUBCYCLE
CLK
T4
T5
T0
T1
T2
T3
T4
T5
T0
T1
80ns
ROUTER CYCLE
LINK UNITS SEND REQUESTS ON LM BUS
REQUEST SELECTOR SENDS GRANT TO ONE LINK UNIT ON LM BUS
SEND PACKET HEADER
ROUTING TABLE ACCESSED
LINK UNITS SUPPLY AVAIL OUTPUT LINE DATA ON LM BUS
ROUTER INPUTS LINK VECTOR AND AVAIL LINES
ROUTER SUBCYCLE
CLK
T03
T4
T5
T0
T1
T2
T3
T4
T5
T0
T1
ROUTER INPUTS LINK VECTOR AND AVAIL LINES
ROUTER OUTPUTS SELECTED LINK MASK ON LM BUS

FIGURE 16

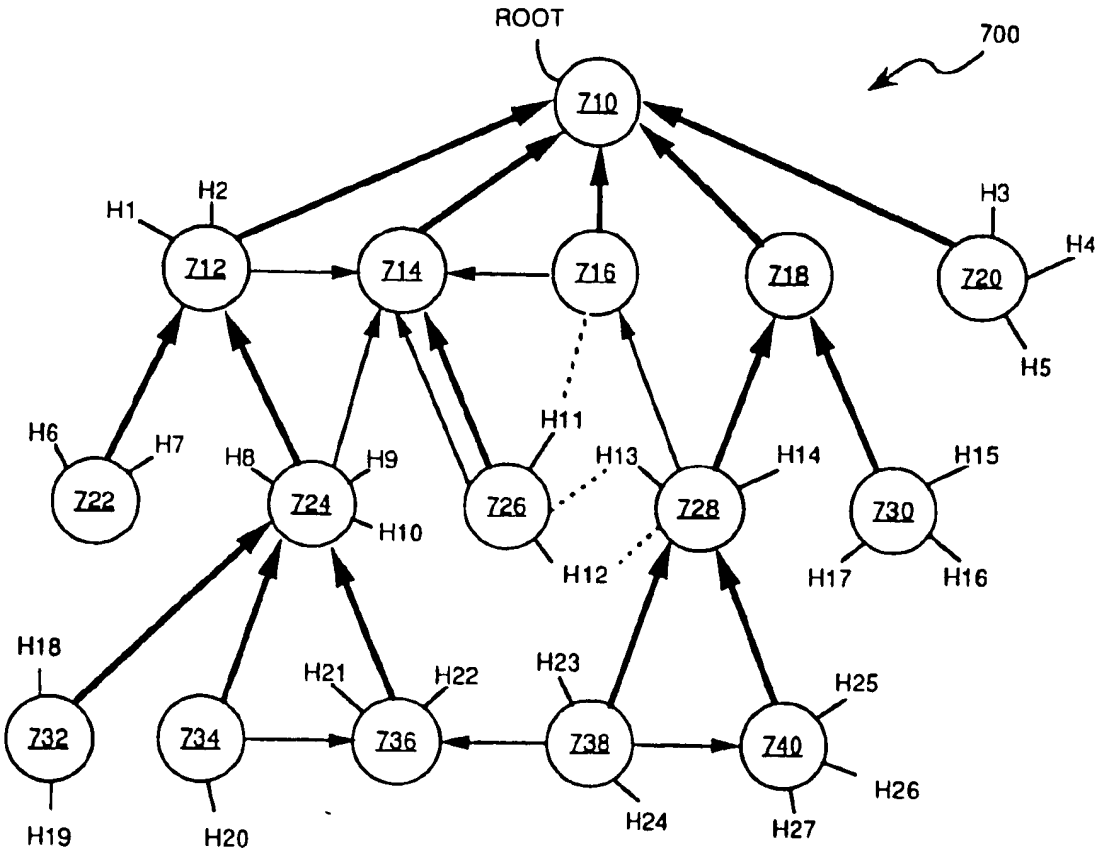

ROOT
700
Z10
H1
H2
Z12
Z14
Z16
Z18
H3
H4
Z20
H5
H6
H7
Z22
H8
H9
Z24
H10
H11
Z26
H12
H13
Z28
H14
H15
Z30
H16
H17
H18
H19
Z32
H20
Z34
H21
H22
Z36
H23
Z38
H24
H25
Z40
H26
H27

FIGURE 17

Martin, Kim

From: Beth Isabelle [isabelle@law.harvard.edu]
Sent: Friday, October 12, 2001 12:24 PM
To: larry.cooperman; EFlynn9823; kmartin; jeananne.piper; BEB930; isabelle
Subject: name change article

for anyone who does not already have it:

Liquidix, Inc. Purchases Advanced Fluid Systems

09/27/2001
PR Newswire
(Copyright (c) 2001, PR Newswire)

PHOENIX, Sept. 27 /PRNewswire/ -- Effective September 12, 2001, the Board of Directors and shareholders of Learner's World, Inc. (OTC Bulletin Board: LRNW) approved a name change to Liquidix, Inc. Thereafter Liquidix, Inc. also recently obtained NASD approval to change its trading symbol from LRNW to LQDX effective September 27, 2001.

Effective September 26, 2001 Learner's World, Inc. purchased Advanced Fluid Systems, Inc. from Liquidics, Inc. through stock and cash with terms to be listed in Liquidix, Inc. 8K filings to be undertaken. Therefore Advanced Fluid Systems, Inc. will now operate as a wholly owned subsidary of Liquidix, Inc. (OTC Bulletin Board: LQDX) Through its recent contracts with several Military/Government Agencies and US National Laboratories including NASA, the company has realized significant growth.

Contact Investor Relations: 1-800-760-1166.

--

Beth A. Isabelle
Mailto:Isabelle@law.harvard.edu